AFRL-OSR-VA-TR-2013-0173

A UNIFIED STEGANALYSIS FRAMEWORK

Nasir Memon
Polytechnic University

April 2013
Final Report

DISTRIBUTION A: Approved for public release.

AIR FORCE RESEARCH LABORATORY
AF OFFICE OF SCIENTIFIC RESEARCH (AFOSR)
ARLINGTON, VIRGINIA 22203
AIR FORCE MATERIEL COMMAND

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 25-02-2013	**2. REPORT TYPE** FINAL REPORT	**3. DATES COVERED** *(From - To)* 01-02-2009 to 30-11-2012

4. TITLE AND SUBTITLE A UNIFIED STEGANALYSIS FRAMEWORK	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER FA9550-09-1-0087
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Memon, Nasir D	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Polytechnic University of NYU, Six MetroTech Center, Brooklyn, NY 11201	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Office of Scientific Research Suite 325, Room 3112 875 N. Randolph Street Arlington, VA 22203-1768	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-OSR-VA-TR-2013-0173

12. DISTRIBUTION/AVAILABILITY STATEMENT

DISTRIBUTION A: APPROVED FOR PUBLIC RELEASE

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Steganalysis research is mainly motivated to investigate the possible features that yields highest accuracy in identifying cover objects subjected to steganographic embedding. Although design of better steganalysis techniques is a crucial goal, given the diversity of steganography techniques and their ever-increasing sophistication, it is not realistic to assume a single technique will outperform others in identifying all types of steganographic techniques. Therefore in practical application scenarios, these competing techniques have to be incorporate together. From this perspective, the most important goal of a steganalysis system, that combines many individual steganalysis techniques, is to improve performance. This project focused on real-world deployment of steganalysis systems. Motivated by this problem, we pursue two fundamental questions: First, how to incorporate several steganalyzers together in a steganalysis system, and second, how to limit the testing cost of a steganalyzer.

15. SUBJECT TERMS

Steganography, Steganalysis, Fusion, Classifiers.

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Nasir Memon
U	U	U	SAR		**19b. TELEPHONE NUMBER** *(Include area code)* 718-260-3970

Reset

AFOSR Project Award No: FA9550-09-1-0087
"Unified Steganalysis Framework"
Final Performance Report for

Nasir Memon
Polytechnic Institute of New York University
Brooklyn, NY

Steganalysis research is mainly motivated to investigate the possible features that yields highest accuracy in identifying cover-objects subjected to steganographic embedding. Although design of better ste- ganalysis techniques is a crucial goal, given the diversity of steganography techniques and their ever-increasing sophisti- cation, it is not realistic to assume a single technique will outperform others in identifying all types of steganographic techniques. Therefore in practical application scenarios, these competing techniques have to be incorporated together. From this perspective, the most important goal of a steganalysis sys- tem, that combines many individual steganalysis techniques, is to improve performance. This project focused on real-world deployment of steganalysis systems.

Motivated by this problem, we pursue two fundamental questions: First, how to incorporate several steganalyzers together in a steganalysis system, and second, how to limit the testing cost of a steganalyzer. To address these questions, essentially, we focused on the machine learning aspect of steganalyzer design and introduce two ensemble based classification systems with distinct advantages, namely a hierarchical ensemble of classifiers based approach and a decision tree based approach. The ensemble of classifiers based system provides a workable and systematic procedure to incorporate several steganalyzers together in a composite steganalyzer to improve detection performance in a scalable and cost-effective manner. The decision tree based system, alternatively, aims at minimizing the computational cost of steganalysis, while still maintaining the detection accuracy.

The underlying ideas for both of the techniques are briefly described in the subsequent sections. The details and the corresponding results can be found in the attached papers.

Research Contributions

Ensemble classifiers are introduced to improve the performance of individual classifiers. An ensemble classifier is a system that makes classification decisions based on the decisions from its base classifiers. We view a steganalyzer as essentially consisting of two main steps. The first step involves identifying a specific set of statistics (more commonly referred to as features) derived from cover-objects that are in general sensitive to steganographic embedding. The second step involves the use of machine learning algorithms to automatically generate classification models from extracted features that can discriminate between cover-objects and steganographically embedded objects, i.e., stego-objects. We extend the ensemble classifier idea into steganalysis design by incorporating it to the machine learning compenent, while utilizing many features associated with different steganalyzers, to improve both accuracy and computational cost of steganalysis.

An Ensemble of Classifiers Approach to Steganalysis

The first systems uses a hierarchical ensemble of classifiers based approach and provides a procedure that utilizes feature vectors from many steganalyzers to build a multi-class classification system which can distinguish stego-objects created by different stegano-graphic methods from each other and from cover-objects. A basic outline of the scheme is displayed in Fig. 1, where each module serves as a binary classification system built by combining many boosted ensembles of classifiers. Essentially, the modules take as input all the features and each module specializes in only detecting one steganography technique. Decisions of the individual modules are later consolidated to make a final decision on the class a given object may belong. The resulting system not only improves detection performance but it also supports incremental addition of new steganalysis and steganography techniques and removal of existing ones by adding, removing and updating modules. Moreover, due its incremental learning capability, when new sets of stego- and cover-objects are available for training, the system is able avoid retraining with the previously used training data by expanding ensembles in each module.

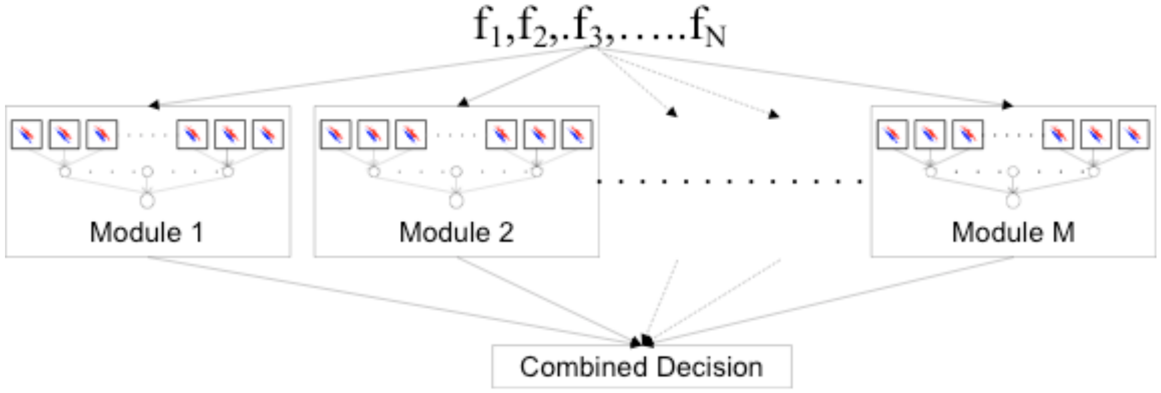


Fig. 1 The structure for hierarchical ensemble of classifiers based steganalysis approach

A Cost-Effective Decision Tree Based Approach to Steganalysis

The second system we present is based on a decision tree approach. This system aims at minimizing the computational cost of classification. To our knowledge, the design of ste-ganalyzers that perform effectively with limited computational power has not been considered in the literature. Instead, feature selection has been usually employed as a pre-processing step to reduce the cost incurred by feature measurement. In practice, it is desirable that the computational cost associated with feature acquisition or measurement should be taken into account when designing cost efficient steganalysis systems. The decision tree based system attempts to achieve this while maintaining the classification accuracy and without performing feature selection. Fig. 2 depicts a simplified diagram of the ensemble tree classifier. A base decision tree classifier and its boosted versions are organized as a meta-classifier. In the resulting decision tree meta-classifier, leaves represent classifications or decisions and internal nodes correspond to features. Starting at the root node, at each level of tree, value of a feature determines the path to be taken until a leaf is reached. Each path from root to leaves (like the one marked in red) is a decision rule and requires computation of a subset of features in sequence as opposed to computing of all features in advance.

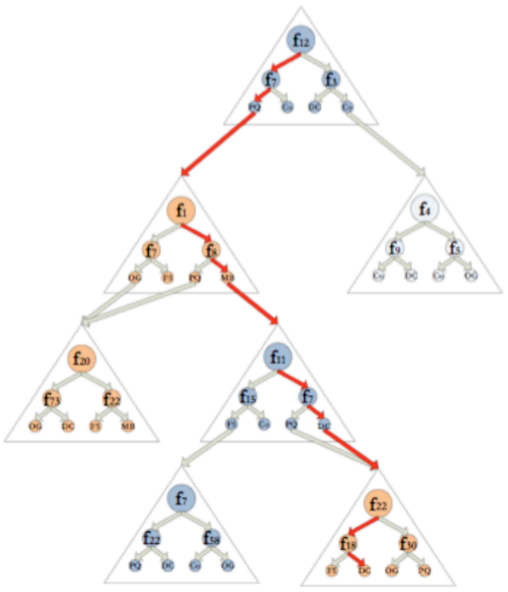


Fig. 2 The structure for decision tree based steganalysis approach.

Results:

To evaluate the performances, both systems are built by combining feature sets associated with well five well-known universal steganalysis methods and tested against four widely used steganography methods.

Experimental results show that hierarchical ensemble classifier is able to improve steganalysis performance as compared to conventional feature level fusion, where a single classifier is trained on joint features formed by stacking all the available features. This system mainly provides a modular and scalable structure as it allows for easy addition and removal of techniques and provides incremental learning ability such that when a new batch of data is available, new base classifiers can be trained and added to existing ensembles. However, it lacks on the computational efficiency aspect since for each test instance all the features have to be calculated and tested on a large number of classifiers.

Results also show that the decision tree based system provides the ability to limit the computational cost (in the online phase) of steganalysis to maintain a comparable accuracy to best classifiers such as SVM. Since decision tree based steganalyzer is an online algorithm which requires values of new features dependent on previously provided feature values, on average, only a few of all the features have to be computed before making a decision. It is observed that this steganalyzer also exhibits characteristics of incremental learning, as it is convenient to add new training data to the system. A limitation of the decision tree based classifier is that when new techniques are added or removed, the system has to be retrained.

Project Outcomes

The project work included both research and implementation. Both legs of the work have involved contributions from three research students, a post-doctoral researcher and a faculty member.The outcome of the research project included three research papers. One part of the completed work is presented in ACM ICPR 2010 and another is accepted for presentation in SPIE-EI 2013 Conference. Third paper is submitted for publication in IEEE Transactions on Information Forensics and Security. We are currently revising the first version by including more recent steganography and steganalysis techniques. A version of initially submitted version of the journal paper is available as a technical report.

An Ensemble of Classifiers Approach to Steganalysis

S. Bayram, A. E. Dirik
ECE Dept., NYU Poly
{sevinc,emir}@isis.poly.edu

H. T. Sencar
CE Dept., TOBB ETU
htsencar@etu.edu.tr

N. Memon
CSE Dept., NYU Poly
memon@nyu.edu

Abstract

Most work on steganalysis, except a few exceptions, have primarily focused on providing features with high discrimination power without giving due consideration to issues concerning practical deployment of steganalysis methods. In this work, we focus on the machine learning aspect of steganalyzer design and utilize a hierarchical ensemble of classifiers based approach to tackle two main issues. Firstly, proposed approach provides a workable and systematic procedure to incorporate several steganalyzers together in a composite steganalyzer to improve detection performance in a scalable and cost-effective manner. Secondly, since the approach can be readily extended to multi-class classification it can infer information from a given stego-object about the steganographic embedding technique. We provide results to demonstrate the potential of the proposed approach.

1. Introduction

Steganography aims to prevent an observer from realizing that any covert communication is taking place. Correspondingly, the objective of steganalysis is to detect such communication and to obtain definitive information on the steganographic technique so that ultimately the hidden information can be extracted. There have been two main research approaches to the problem of steganalysis, namely, technique-specific steganalysis and universal steganalysis. The former group of techniques perform very accurately when used against the steganographic technique it is targeted for. The latter group of techniques, on the other hand, are effective over a wide range of techniques, while performing less accurately overall. However, since universal steganalysis is better suited to the practical setting, it attracted more interest and many effective steganalyzers are proposed.

[0]The work on this paper was supported by the AFOSR grant number FA9550-09-1-0087

The research on universal steganalysis has primarily focused on identifying features that can better discriminate cover-object from stego-object without much consideration to other critical aspects. First aspect is due to lack of a systematic treatment to integrate various steganalyzers, specific and universal, together in a composite steganalyzer to improve achievable detection performance. Such a composite steganalyzer need to be scalable by allowing for easy addition of new steganalyzers and removal of existing steganalyzers. Further, training with newly available data should be as cost effective as possible. The other important aspect is that such a steganalyzer should not only reliably differentiate between cover-object from stego-object but must also be able discriminate different steganographic techniques from each other so that more elaborate analysis of the stego-object can be possible.

In the literature, there are a limited number of studies that partially address above goals. Kharrazi et al. [5] to improve accuracy of steganalysis proposed a composite steganalyzers built using a number of pre-classification and post-classification information fusion strategies. Similarly, in [9, 11], authors proposed a multi-class steganalysis technique to classify stego-object to known steganographic techniques by combining many binary classifiers under one-vs-one strategy. In this paper, we investigate the applicability of a hierarchical ensemble of classifiers based approach to fuse information from different steganalyzers and to achieve above goals [12, 8]. The resulting steganalyzer also provides incremental learning capability and it is inherently suited to multi-class classification scenarios. In the following section, we describe the details of the proposed steganalyzer. Details of the test dataset and experimental results are reported in Sections 3 and 4, respectively.

2. Ensemble Based Steganalyzer

Proposed composite steganalyzer utilizes feature vectors from many steganalyzers to build a decision framework. It is essentially a multi-class classification system that can distinguish stego-object created by different stegonagraphic methods from each other and from cover-object. For each class, we build a binary classifier that makes a soft decision as to whether a given media object belongs to that particular class using a one-vs-rest scheme. Each binary classifier fuses the decision of different steganalyzers, and to ensure improved performance, as compared to individual steganalyzers, a boosting approach is used as learning method.

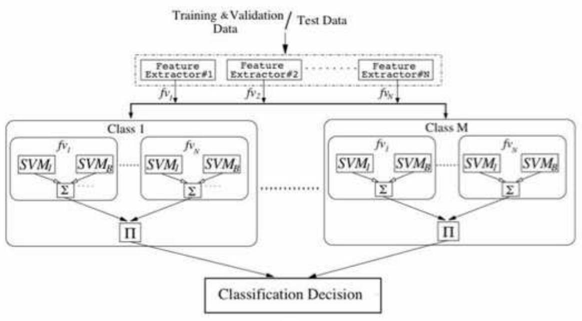


Figure 1. Layout of the composite steganalyzer.

Figure 1 shows the layout of the proposed hierarchical ensemble of classifiers based construction. In this system, objects are assumed to belong one of $Class_i$, $i = 1, 2, \ldots, M$, where M is the total number of classes, and each class is associated with either one of the embedding techniques or the cover-object set. (This configuration can be trivially reduced to a binary classification problem by combining all embedding techniques in the same class.) Here, each steganalyzer is represented by its *feature vector* , denoted by fv_j, $j = 1, 2, \ldots, N$ where N is the total number of steganalyzers. In each class, using boosting, an ensemble of classifiers is trained on each individual feature vector, as opposed to a single classifier, to yield a stronger classifier with higher detection accuracy [2]. Each ensemble is generated by combining a total of B base-classifiers, which is selected to be a support vector machine (SVM), through a three-step iterative procedure.

- A binary base-classifier is trained on a subset of training data which is selected according to a probability distribution that assigns weights to all data points in the training data and adaptively adjusts them at each step. This is followed by verifying the trained base-classifier's performance on a separate validation data.
- Most recently trained base-classifier is combined together (by a weighted sum rule that takes into consideration individual error rates) with the all previous trained ones to ensure that the overall error rate on training data is acceptable. The weights of training data is re-adjusted by amplifying the weights of the mis-classified training data (so that in the next step the subsequent base-classifiers focus more on the mis-classified data).
- When all the B base-classifiers are trained, overall error of the combined classifier is computed over validation data and then translated into a metric to determine the reliability of this binary classifier.

This process essentially determines how reliable each feature vector is in discriminating objects belonging to a specific class. Then, each ensemble of classifiers, associated with different feature vectors, are further combined together by a weighted product rule to form the binary classifier. The outputs of each binary classifier is later combined together to make a final decision to determine the class of an object. It must be noted that the binary classifier does not make a hard decision to determine the membership of the object in that class but rather generates a probability value to indicate its confidence in its decision. The overall decision is made by evaluating the outputs of each binary classifier.

Resulting composite steganalyzer has the following properties.

1. The improvement in performance is two-fold. First, it ensures that, due to underlying boosting methodology, for each feature vector a strong classifier with near-optimal error performance is constructed. Second, unlike fusing at the feature vector level or at the decision level, the composite steganalyzer organize feature vectors with respect to their discrimination power in each class.
2. When new steganalysis techniques are to be added to the composite steganalyzer, an ensemble of base classifiers is trained with the training data and combined with existing binary classifier previously designed for each class.
3. Addition of new a steganographic method essentially requires generating a new binary classifier (i.e., training an ensemble of base-classifier for each steganalyzer and combining them) and incorporating it with the composite steganalyzer without interfering with the existing binary classifiers. Removal of a steganographic method, on the other hand, just requires discarding the corresponding binary classifier.
4. When a new batch of training data is available, the composite steganalyzer is updated by adding new base-classifiers to the ensemble. Since existing base-classifiers are already trained on the previously seen training data, the new base-classifiers can be trained on the unseen training data before being incorporated into the composite steganalyzer.

5. The ability to determine what classes an object is most likely to belong can be utilized in either extraction of the message or inferring information about the specifics of the embedding technique, e.g, embedding domain, location of embedded information and embedding rate.

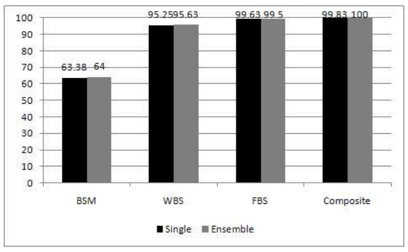
3. Test Dataset

In the experiments, we used publicly available Greenspun's image dataset which contains more than 1800 images of different scenes. In the experiments, we used four JPEG based steganography techniques: Outguess [13], F5 [16], model based embedding [14], and perturbed quantization embedding [4]).

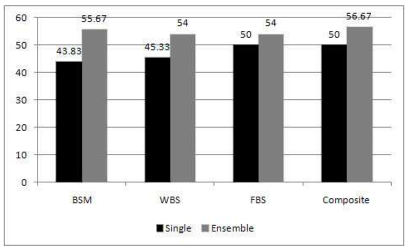
As cover images, we used the images in dataset without any processing on them. We also compressed these images again since some of the steganography methods are double compressing the images. Stego-images are generated by embedding randomly chosen messages (in bits) into 1600 grayscale images using each of the four steganography techniques. A random message length was determined adaptively for all images based on their number of non-zero DCT (NZ-DCT) coefficients. Embedding rate is determined by fixing the ratio of message length to the number of NZ-DCT coefficients of the cover-image. Selected embedding rates for Outguess, F5, model based embedding, and perturbed quantization embedding are 0.2, 0.1, 0.4, and 0.2 bits per NZ-DCT, respectively. The composite steganalyzer is built using 5 universal steganalysis techniques: Binary Similarity Measure (BSM) [1], Wavelet Based Steganalysis (WBS) [7], Feature Based steganalysis (FBS) [3], Merged DCT and Markov Features (MRG) [10, 15], Joint Density Features (JDS) [6].

4. Experimental Results

We conducted several experiments to show the effectiveness of the proposed composite steganalyzer under different scenarios. In all experiment, we used 500 images for training, 500 for validation and 600 for testing. In our first experiment, our aim was to show how the boosting based ensemble of classifiers improve performance when compared to a single classifier. For this purpose, we first focused on designing binary classifiers to discriminate cover-images from Outguess embedded stego-images. Three features vectors are extracted from cover and stego-images using BSM, WBS, and FBS steganalyzers. We utilized each feature vector a single SVM classifier and in addition, an ensemble based classifier by boosting three SVM classifiers, $B = 3$, is designed. The performance of these steganalyzers are given in Figure-2(a). Same experiment is carried for PQ as well, and corresponding results are given Figure-2(b). Results show that there is only a slight increase in the performance with boosting. This is mainly because boosting is effective in building a strong classifier as a combination of many weak classifiers and SVM cannot be considered a weak classifier. In addition, we generated a binary composite steganalyzer by fusing the decision of ensemble classifiers as explained in Section 2. We also designed a single classifier by combining all the feature vectors into one feature vector. The comparison of the two fusion strategies are given in the last columns of Figure-2(a) and Figure-2(b). Since for Outguess individual steganalyzers perform satisfactorily, the improvement due the composite steganalyzers is marginal; however, on PQ technique it is considerable.



(a) Steganalysis of Outguess



(b) Steganalysis of PQ

Figure 2. Performance comparison of single and ensemble based system.

In the second experiment, we tested the performance of composite steganalyzer by combining the above 3 feature vectors under a five-class classification scenario where 4 out of 5 classes are associated with steganographic techniques, and the fifth class is due to cover-images (half single, half double compressed). Table-1 gives the confusion matrix for the designed composite steganalyzer. It can be verified that the errors mainly occurred due to cover objects being mis-identified as PQ objects (which is consistent with the results of our first experiment).

In the last experiment, we introduced new feature vectors to the system to see the change in the performance. For this purpose, JDS and MRG feature vec-

Table 1. Performance Results on Overall system with BSM, WBS and FBS

	Cover	F5	MB	OG	PQ
Cover	0.76	0.002	0.058	0	0.18
F5	0	0.998	0	0.002	0
MB	0	0	0.998	0.002	0
OG	0	0	0	1	0
PQ	0.136	0	0.022	0.002	0.84

tors are also extracted from all the cover- and stego-images. Ensemble of 3 base-classifiers (SVMs) are generated and these ensembles are then added to the existing composite steganalyzer. Detection accuracy of the composite steganalyzer based on 3 steganalyzers (BSM, WBS,FBS) and the one based on 5 steganalyzers (BSM, WBS, FBS, JDS, MRG) are given in Figure-3 for comparison. It can be inferred from this results that the added feature sets improved composite steganalyzer‘s ability to achieve 99.83% accuracy despite the increase in the dimension of feature space from 113 features to a total of 556 features.

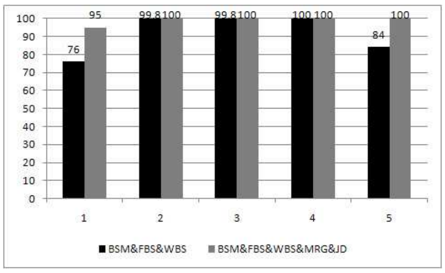


Figure 3. Comparison of detection accuracy of the two composite classifiers.

5. Conclusions

The design procedure of most steganalyzers can be essentially reduced to two basic tasks: identifying a set of distinguishing features to reliably discriminate cover-objects from stego-objects and building a machine learning algorithm for classification. Focusing on the machine learning aspect, in this work, we introduced an alternative steganalysis decision framework that utilizes a hierarchical ensemble of classifiers. The resulting construction not only allows fusing many steganalysis techniques and provides inherent support for multi-class classification (detection of steganograhy techniques) but also is scalable and incrementally updatable. The efficacy of the proposed framework was tested with the fusion of up to 5 popular steganalysis schemes on 4 JPEG based steganography methods concerning different scenarios. The experiments show that significant improvements on detection accuracy are achieved in all cases, most notably, the composite steganalyzer based on 5 steganalyzers is observed to have %99 detection accuracy.

References

[1] I. Avcıbas, M. Kharrazi, N. Memon, and B. Sankur. Image steganalysis with binary similarity measures. *EURASIP Journal on Applied Signal Processing*, 17:2749–2757, 2005.

[2] Y. Freund and Y. E. Schapire. A decision-theoretic generalization of on-line learning and an ap- plication to boosting. *Journal of Computer and System Sciences*, 1997.

[3] J. Fridrich. Feature-based steganalysis for JPEG images and its implications for future design of steganographic schemes. In *Information Hiding, 6th International Workshop*. Springer, 2004.

[4] J. Fridrich, M. Goljan, and D. Soukal. Perturbed quantization steganography. *Multimedia Systems*, 11(2):98–107, 2005.

[5] M. Kharrazi, H. T. Sencar, and N. Memon. Improving steganalysis by fusion techniques: A case study with image steganography. *LNCS Transactions on Data Hiding and Multimedia Security*, 4300:123–137, 2006.

[6] Q. Liu, A. Sung, and M. Qiao. Improved detection and evaluation for JPEG steganalysis. In *Proceedings of the seventeen ACM international conference on Multimedia*, pages 873–876. ACM, 2009.

[7] S. Lyu and H. Farid. Steganalysis using higher-order image statistics. *IEEE Tran. on Information Forensics and Security*, 1:111–119, 2006.

[8] D. Parikh and T. Chen. Data fusion and cost minimization for intrusion detection. *IEEE Tran. on Information Forensics and Security*, 3:381–389, 2008.

[9] T. Pevny and J. Fridrich. Multiclass blind steganalysis for jpeg images. *Proc. of SPIE*, 6072:1–13, 2006.

[10] T. Pevny and J. Fridrich. Merging Markov and DCT features for multi-class JPEG steganalysis. *Security, Steganography, and Watermarking of Multimedia Contents IX*, pages 1–13, 2007.

[11] T. Pevny and J. Fridrich. Multi-class detector of current steganographic methods for jpeg format. *IEEE Tran. on Information Forensics and Security*, 3:635–650, 2008.

[12] R. Polikar. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*, 6:21–45, 2006.

[13] N. Provos. Defending against statistical steganalysis. In *10th USENIX Security Symposium*, volume 10, pages 323–336. Citeseer, 2001.

[14] P. Sallee. Model-based steganography. *Lecture Notes in Computer Science*, pages 154–167, 2003.

[15] Y. Shi, C. Chen, and W. Chen. A Markov process based approach to effective attacking JPEG steganography. *Lecture Notes in Computer Science*, 4437:249, 2007.

[16] A. Westfeld. F5-a steganographic algorithm: High capacity despite better steganalysis. In *Information Hiding: 4th International Workshop, IH 2001, Pittsburgh, PA, USA, April 25-27, 2001: Proceedings*.

A Cost-Effective Decision Tree Based Approach to Steganalysis

Liyun Li[a], Husrev Taha Sencar[b] and Nasir Memon[a]

[a]Polytechnic Inst. of NYU, 6 Metrotech Center, Brooklyn, NY, USA;
[b]TOBB University of Economics and Technology, Ankara, TURKEY

ABSTRACT

An important issue concerning real-world deployment of steganalysis systems is the computational cost of acquiring features used in building steganalyzers. Conventional approach to steganalyzer design crucially assumes that all features required for steganalysis have to be computed in advance. However, as the number of features used by typical steganalyzers grow into thousands and timing constraints are imposed on how fast a decision has to be made, this approach becomes impractical. To address this problem, we focus on machine learning aspect of steganalyzer design and introduce a decision tree based approach to steganalysis. The proposed steganalyzer system can minimize the average computational cost for making a steganalysis decision while still maintaining the detection accuracy. To demonstrate the potential of this approach, a series of experiments are performed on well known steganography and steganalysis techniques.

Keywords: computational cost effective, decision tree, steganography

1. INTRODUCTION

In recent years, there have been a considerable increase in the research efforts to develop better steganalysis techniques. As a result, both the design of steganalysis tools (steganalyzers) and their ability to accurately identify steganographic communication witnessed signification improvements. In its initial phase, steganalysis techniques were primarily developed by targeting specific steganographic techniques.[1–4] This approach yielded techniques that perform very accurately when used against the targeted techniques. Nevertheless, the approach is not suited to the practical setting of steganalysis as it implicitly assumes the knowledge of steganography technique used for embedding.

To be functional over a wide range of techniques, later steganalysis techniques adopted a more general approach by essentially focusing on statistical properties of cover data, i.e., cover-objects used as the host for embedded information and identifying those which are likely to exhibit significant change after embedding operation. This approach to steganalyzer design is often referred to as universal or blind steganalysis. Although universal steganalysis techniques perform less accurately, they can potentially identify most steganographic techniques with very limited prior information on steganographic techniques.[5–11]

Although designing accurate steganalysis techniques is very important, deployment of these techniques as part of a steganalysis system involve many other critical issues that are not yet sufficiently addressed. Consider a system that has to perform steganalysis on a very large amount of data using fixed computational resources under real-time or near real-time constraints. Given all prominent steganalyzers utilize statistical machine learning to achieve good levels of performance, it is important that their design can respond to such a challenge. Essentially, during the online (test) phase, computational complexity of steganalysis will depend on the amount of computation that needs to be performed when extracting features from a given object. Therefore, the ability to limit the cost of acquiring features is the key to address this challenge.

In this paper, we focus on the computation cost of acquiring stego-feature values when applying machine learning classifiers to perform steganalysis. Till now, design of steganalyzers that perform effectively with limited

Further author information: (Send correspondence to Nasir Memon)
L. Li: E-mail: liliyun02@gmail.com
H. T. Sencar: E-mail: htsencar@etu.edu.tr
N. Memon: E-mail: memon@nyu.edu

computational budget has not been considered; instead, feature selection is usually employed as a pre-processing step to reduce the cost incurred by feature measurement. In practice, it is desirable that the computational cost associated with feature acquisition or measurement should be taken into account when designing steganalysis systems. In scenarios where the computational cost for acquiring feature values is high, it is more crucial to deploy a steganalyzer that takes into account the acquisition cost of each feature. Motivated by this, we propose a decision tree based steganalysis system, which aims at minimizing the computational cost of classification while still maintaining satisfactory accuracy. The proposed system has the capability to judiciously choose different sets of features cost effectively for different stego-objects. Therefore on average, computational cost for steganalysis is optimized.

2. THE DECISION TREE BASED COST EFFECTIVE APPROACH FOR STEGANALYSIS SYSTEMS

Steganalyzers essentially differ in the features they use to differentiate between stego- and cover-objects. Consequently, each steganalyzer performs differently in identifying different steganographic embedding techniques. When building a real world steganalysis systems, therefore, an important concern is how to combine various steganalyzers together so that an overall much stronger steganalyzer can be built. In our approach, we view each steganalyzer as a decision tree classification system and try to build a composite steganalyzer, by combining decision trees associated with different steganalyzers together, while at the same time reducing feature acquisition cost. The approach described in this section applies a meta-classifier CoCoST[12] which combines individual decision trees of different steganalysis features together. To achieve this, our approach applies a meta-classifier CoCoST[12] that makes it possible to combine individual decision trees in a very effective manner.

Decision trees (DT) are widely used in machine learning, and there are two attracting reasons to utilize decision trees in steganalysis. First, decision tree classifiers have accurate predictive capabilities, and they are easy to train. Second, prediction process in a decision tree is inherently incremental, which makes it computationally very efficient. In a decision tree, as long as the the path for classifying a given stego-object does not include all the features, the number of features needed to label this object (as stego- or cover-object) is reduced. Hence, for decision trees to make predictions, we do not need to provide all the feature values beforehand. Instead, we can extract features only when the encountered decision tree node requires a certain feature value. This is important because it not only exempts us from pre-computing all the feature values for each steganalyzer, but also reduce the average number of features needed to detect a stego-object.

This characteristics of decision trees is very desirable when there is a cost associated with acquiring each feature (which is the usual case in steganalysis). We define the expected cost for making a decision for an unknown random object as the expected cost of a decision tree. For a specific stego-object, its prediction cost is the total cost for extracting or acquiring all the feature values for classifying this object along its root-to-leaf path.

2.1 Base Steganalyzer: the Individual LASC Decision Tree

The LASC (Looking Ahead Suppressed Cost) decision tree is the base classifier of the CoCoST steganalysis system. There are two key characteristics of the LASC algorithm. The first one is that LASC utilizes not only the entropy gain, but also the size of the node where the size of the node is the proportion of training instances in each node. For example, the root node has a size of 1, while the leaf nodes has much smaller size. None of those cost-efficient variants of C4.5 trees take this information into consideration. However, at the root node, where the node size is big and every instance gets tested against the root feature, we want the root feature to be very cheap and efficient. On the contrary, at the leaf nodes where few instances enter, we can tolerate using more number of features and features with expensive cost for a better discrimination power. The LASC heuristic is expressed as

$$H = \frac{\triangle I}{freq^{\alpha}C + (1 - freq^{\alpha})}. \tag{1}$$

. In Eq. 1, $\triangle I$ is the *look ahead* entropy gain which represents the information we obtained by choosing this feature to separate the data; C is the cost associated with each feature. $freq$ denotes the size of the node which is the proportion of training instances at that node and the constant parameter α designates the sensitivity of the heuristic to cost.

The core idea the LASC heuristic is that the sensitivity to cost is suppressed as the tree grows and the size of the node gets smaller. Consider the root node whose size is 1. The root node is a big node and each instance has to be tested against the root feature. Then, $1 - freq^{\alpha}$ will be around zero and the term $freq^{\alpha}C$ will be dominating. Therefore, the heuristic will choose the most cost-efficient feature. As the tree is expanded and the nodes become smaller, fewer data instances will enter to those lower level nodes at which we may deploy more discriminative features. In this case, the term $freq^{\alpha}$ gets smaller and the term $1 - freq^{\alpha}$ dominates. Correspondingly, the heuristic now picks the feature that provides the most information. In summary, a LASC heuristic yields an efficient decision tree by reducing the average number of features used for classification.

The other key characteristic of LASC tree is that when calculating the entropy gain, it allows looking ahead one or more steps (k steps) by looking for a better combination of features instead of looking for individually good features when calculating the heuristic and building the tree. It should be noted that there is trade-off between building better trees and the training time as training takes more time with further look-ahead. As the computational cost of several features may depend on each other, the Look-Ahead heuristic will be able to choose the most fitting set of dependent features by considering their dependent cost together. The LASC tree is constructed recursively by selecting the most fitting feature according to the heuristic score.

2.2 Inverse Boosting to Diversify the Pool

CoCoST classifier utilizes standard AdaBoost[13] as well as inverse boosted decision trees as its base classifiers. Inverse boosting was originally introduced to create ensembles of classifiers with high varieties[14].[15] Instead of raising the weight of misclassified instances, we raise the weight of correctly classified instances, and the updating function for the new weight is the same as standard boosting but with the sign flipped for the indicator. Given the weights $D_t(i), i = 1, 2, ..., M$ and the error rate ϵ_t of the current classifier, updated weights for each inverse boosting round is obtained as

$$D_{t+1}(i) = \frac{D_t(i)exp(\beta_i I(y_t(i) = h_t(\vec{X}_i))}{Z_t}, \tag{2}$$

where Z_t denotes the normalizing vector, I is the indicator function which raises the weight when the instance is misclassified and reduces it when the instance is correctly classified, i.e,

$$I(y_t(i) = h_t(\vec{X}_i)) = \begin{cases} 1, & \text{if } y_t(i) = h_t(\vec{X}_i), \\ -1, & \text{otherwise;} \end{cases} \tag{3}$$

and β is a parameter calculated from the error rate ϵ as

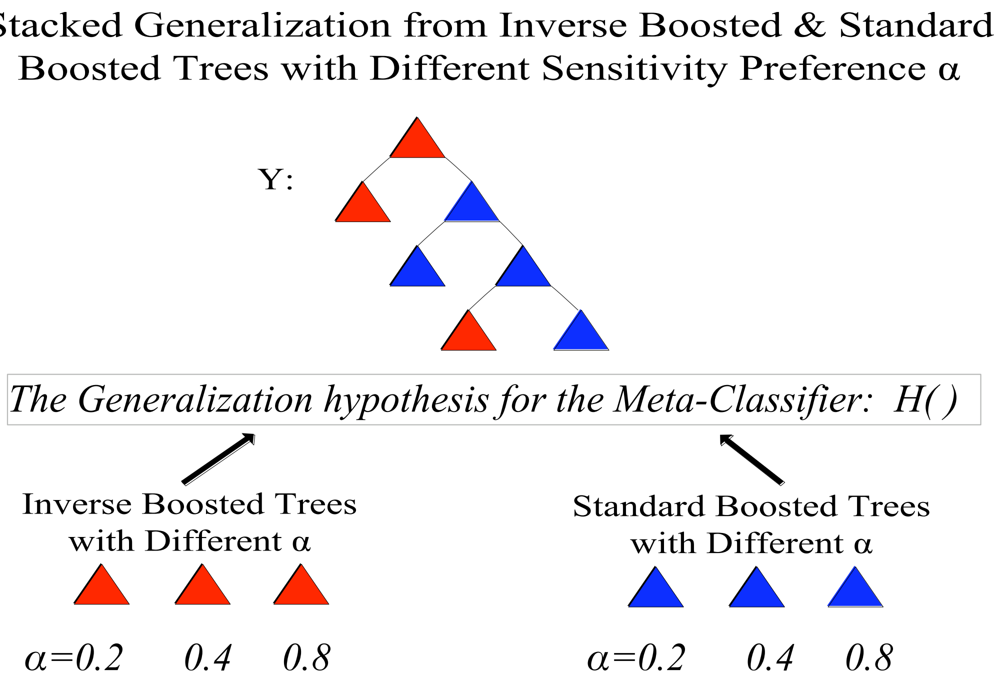
$$\beta_t = \frac{1}{2} log \frac{1 - \epsilon_t}{\epsilon_t}. \tag{4}$$

By performing boosting and inverse boosting, we obtain a more diversified pool of classification trees. It should be noted again that standard boosted trees tend to focus on instances that are hard to classify, while inverse boosted trees are cheaper and tend to focus on easy instances using fewer features. It is worth mentioning that depending on what speed we want the boosting performance to converge, the weights can be updated more aggressively or conservatively as compared with Eq. 2.

2.3 Create Meta-Classifier: Combining the Trees by Stacked Generalization

The pool of steganalyzer trees obtained by boosting and inverse boosting, are incorporated together to obtain the CoCoST classification system. This is realized by using a method called stacked generalization. In stacked generalization, each boosted tree predicts on a validation data set and their decisions are treated as the new feature set to build a meta-classifier. The resulting CoCoST meta-classifier is also a tree as shown in Figure 1.

With the decision tree based steganalysis system during the online testing phase, the system will ask for feature values incrementally until a decision can be made. Therefore, there is no need to pre-compute any


Stacked Generalization from Inverse Boosted & Standard Boosted Trees with Different Sensitivity Preference α
Y:
The Generalization hypothesis for the Meta-Classifier: H( )
Inverse Boosted Trees with Different α
Standard Boosted Trees with Different α
α=0.2
0.4
0.8
α=0.2
0.4
0.8
Figure 1. Structure of the CoCoST Classifier

features. This structure also possesses the ability of incremental learning. When fresh data or new steganalysis techniques, i.e., new features, become available, one can train new base decision trees using these new data and features, and then incorporate those newly built trees into the whole system by retraining the meta-classifier, thereby avoiding the overall re-training of the system by keeping the other previously generated trees untouched.

3. EXPERIMENTAL RESULTS

3.1 Studied Techniques

To test the performance of the proposed steganalysis systems, We consider four universal steganalysis techniques and four steganographic techniques. Below we list four JPEG based steganography techniques and briefly state their basic assumption in limiting the embedding distortion they induce on a cover-object. All these techniques embed data in DCT domain without any noticeable visual artifacts.

- *Outguess*[16] tries to preserve DCT coefficient histograms by selecting DCT coefficients that yield least embedding distortion. Outguess re-compresses cover-images prior to embedding at JPEG quality factor 75.
- *F5*[17] embeds by decrements the absolute value of randomly chosen DCT coefficients by one. It employs the notion of 'matrix embedding' to minimize the number of changes made to DCT coefficients. Similar to Outguess, F5 re-compresses cover-images at JPEG quality factor 80.
- *Model Based (MB) Steganography*[18] uses a parametric model of DCT coefficient distributions, obtains model parameters, and then performs embedding in a way that preserves those models. It does not perform re-compression on cover-images.
- *Perturbed Quantization (PQ) Steganogprahy*[19] is based on an information reduction operation like re-compression. During compression, quantization for a certain subset of DCT coefficients is slightly perturbed to embed message bits. The perturbation is by modifying the coefficient so that it quantizes to designated quantization level. Since, stego-images generated by PQ technique very closely mimic double compressed images, it is significantly less detectable than other techniques.

Our composite steganalysis system utilizes feature vectors introduced by the following steganalysis techniques.

- *Binary similarity measures (BSM)*[5] refers to features extracted in spatial domain. The intuition behind the steganalysis technique is that the embedding operation distorts the correlation between bit planes of an image. The change in bit plane correlations are detected through 18 features extracted from seventh and eighth bit planes of cover- and stego-images.
- *Wavelet based steganalysis (WBS)*[7] extracts features in wavelet transform domain. To detect steganographic embedding, WBS aims at capturing underlying statistics of wavelet sub-band coefficients. For this purpose, 72 features are extracted from transformed intensity of the color channels. These 72 features comprise first order statistics (mean, variance, skewness, kurtosis) of sub-band coefficients and higher order statistics computed from linear prediction error.
- *Merged DCT and Markov features* are obtained by combining two sets of features. In,[20] authors proposed the use of a process called calibration to detect stego-images. Calibration is used to estimate macroscopic properties of the cover-image from the stego-image. Then, 23 spatial and DCT domain features are used to capture the difference between an image and its calibrated version. DCT features are then obtained by extending these features to a larger set of 193 features[20] . In a similar manner,[21] modeled the differences between absolute values of neighboring DCT coefficients as a Markov process and computed 81 features from resulting probability transition matrices. To improve the steganalysis accuracy,[20] merged the 193 extended DCT features with the Markov features which resulted with 274 calibrated features. The steganalysis technique based on merged features is reported to have significantly better detection accuracy than other universal steganalysis techniques.
- *Joint Density features*[22] are proposed to capture traces of embedding in terms of relations between DCT coefficients of neighboring blocks. Features are obtained from joint distribution of inter and intra block DCT coefficients. Here we solely utilize 169 intra block joint density DCT features due to their superior detection capabilities.

3.2 Dataset

For the experiments we used publicly available Greenspun's image set that contains 1800 images taken under varying conditions. Prior to steganographic embedding, black borders around the images are removed and all images are converted into grayscale and re-saved in JPEG format wit quality set to 100. Out of all the images, the ones with significantly low embedding capacity (i.e., low number of nonzero DCT coefficients) are discarded which left us with 1600 images. Then four classes of stego-images are created by embedding all the 1600 images with each of the four steganography methods separately. For each embedding technique a different subset of Greenspun's images were used. For embedding, random binary strings are generated as messages. Message lengths are determined adaptively for all images depending on the number of non-zero DCT (NZ-DCT) coefficients in the image. For example, 0.10 BPNZ-DCT (bits per non zero DCT coefficient) embedding rate corresponds to a length in bits that is equal to 10 % of NZ-DCT coefficients of the cover-image. The class of cover-images are then subjected to double compression operation to create the final class of objects. Hence, a total of six classes of images, (i.e., stego-image classes created by Outguess, F5, MB steganogprahy, PQ steganography techniques and double compressed (DC) cover-images, and singly compressed cover-images), are generated and the two steganalysis systems will be tested against these classes to determine their discrimination accuracy.

3.3 Experimental Results

We conducted several experiments to determine the effectiveness of the proposed steganalysis approaches. The goal is to distinguish how well the two systems are able to identify the class of an unlabeled images from among six possible classes. When building the two systems we used 500 images for training, 500 for validation, and remaining 600 for testing.

Using the 500 training images for each class, we built a LASC tree and then applied boosting and inverse boosting to generate multiple decision trees. The steganalysis accuracy of a single LASC tree, CoCoST with 5 trees and CoCoST with 11 trees in discriminating one class of images from all other classes are shown in the bar

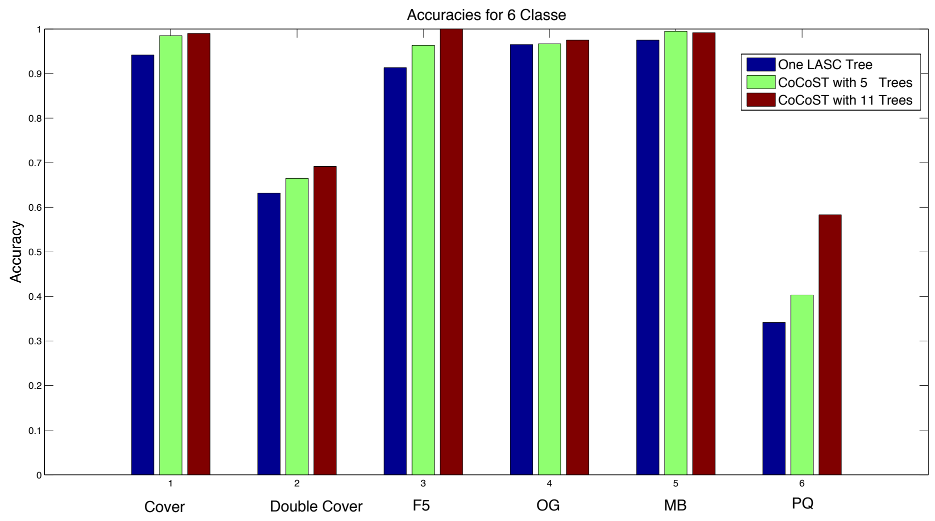


Figure 2. Accuracies for the 6 Classes Using CoCoST Classifier with Different Number of Trees

Table 1. Confusion Matrix of CoCoST with 11 trees

	Cover	Double Cover	F5	Outguess	MB	PQ
Cover	594	3	0	0	0	3
DC Cover	0	415	8	6	0	179
F5	0	0	600	0	0	0
Outguess	7	0	4	585	0	4
MB	2	0	0	0	595	3
PQ	25	219	3	2	1	350

graph given in Figure 3.3. It can be seen that except for DC Cover and PQ steganography classes, increasing the number of trees quickly saturates the accuracy to almost 95%. However, for the two classes increasing the number of boosted trees further do not yield an improvement in classification as expected.

The confusion matrix for 3600 test instances are shown in the Table 1. The total accuracy for all the 6 classes are 86.92%, and the expected number of features used for each instance is 211.56 features, which is 60% less than the features that need to be acquired for each instance in the conventional steganalsis. Also, we see that except for DC Cover and PQ classes which we can only differentiate with an accuracy of 62.92%, the other classes are classified with an accuracy close to 95%. The reason for inferior performance in detecting DC Cover and PQ steganography is that the features are not sufficiently discriminative. Even by adding more boosted trees and focusing on those two classes, accuracies do not increase much. In contrast, the detection accuracy for other classes increase significantly with more boosted trees.

The CoCoST classifier achieves an accuracy of around 88% while it significantly reduced the average number of features used for each data instance by 60%. In a scenario, where we want quick prediction and prefer to extract as few features as possible, the cost-efficient CoCoST classifier is a good choice. However, if our computing power can bear acquiring all the feature values and we want to get the best accuracy out of all the features, we may want to use a more complicated classifier, for example SVM, to get an even better accuracy.

4. DISCUSSION AND CONCLUSIONS

In this paper, we study the steganalysis problem from a practical deployment point of view. More specifically, we focus on optimizing the average computational cost for acquiring different feature values associated with a steganalysis technique. The advantage of the proposed decision tree based steganalyzer is its computational efficiency while still being able to perform as accurate as best classifiers such as SVM. The computational efficiency is inherently obtained because of the tree structure classification process. The decision tree based steganalyzer is an online algorithm which requires values of new features dependent on previously provided feature values. Therefore, on average, only a few of all the stego features have to be computed to make the stego detection. It is shown through experiments that the proposed approach yield viable steganalyzer systems.

REFERENCES

[1] Dumitrescu, S., Wu, X., and Wang, Z., "Detection of lsb steganography via sample pair analysis," in [*Information Hiding*], Petitcolas, F. A. P., ed., *Lecture Notes in Computer Science* **2578**, 355–372, Springer (2002).

[2] Fridrich, J. J., Goljan, M., Hogea, D., and Soukal, D., "Quantitative steganalysis of digital images: estimating the secret message length," *Multimedia Syst.* **9**(3), 288–302 (2003).

[3] Ker, A. D., "Steganalysis of lsb matching in grayscale images," *IEEE Signal Processing Letters* , 441–444 (2005).

[4] Fridrich, J., Soukal, D., and Goljan, M., "Maximum likelihood estimation of secret message length embedded using pmk steganography in spatial domain," *In Proc. SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VII* **5681**, 596–606 (2005).

[5] Avcibas, I., Kharrazi, M., Memon, N., and Sankur, B., "Image steganalysis with binary similarity measures," *EURASIP Journal on Applied Signal Processing* **2005**(17), 2749–2757 (2005). doi:10.1155/ASP.2005.2749.

[6] Avcibas, I., Memon, N. D., and Sankur, B., "Steganalysis using image quality metrics.," *IEEE Transactions on Image Processing* **12**(2), 221–229 (2003).

[7] Lyu, S. and Farid, H., "Steganalysis using higher-order image statistics," *IEEE Transactions on Information Forensics and Security* **1**(1), 111–119 (2006).

[8] Fridrich, J. J., "Feature-based steganalysis for jpeg images and its implications for future design of steganographic schemes," in [*Information Hiding*], Fridrich, J. J., ed., *Lecture Notes in Computer Science* **3200**, 67–81, Springer (2004).

[9] Zhang, T., Li, W., Zhang, Y., Zheng, E., and Ping, X., "Steganalysis of lsb matching based on statistical modeling of pixel difference distributions," *Inf. Sci.* **180**, 4685–4694 (December 2010).

[10] Buml, R., Huber, J. B., and Kaup, A., "Steganographic system based on higherorder statistics," in [*in Proc. SPIE Vol. 5020, Security and Watermarking of Multimedia Contents V*], 156–166 (2003).

[11] Lyu, S. and Farid, H., "Steganalysis using color wavelet statistics and one-class support vector machines," in [*SPIE Symposium on Electronic Imaging*], (2004).

[12] Li, L., Topkara, U., Coskun, B., and Memon, N., "Cocost: A computational cost sensitive classifier," in [*ICDM'09: IEEE International Conference on Data Mining, Miami FL*], (2009).

[13] Dietterich, T. G., "An experimental comparison of three methods for constructing ensembles of decision trees," in [*Bagging, boosting, and randomization. Machine Learning*], 139–157 (2000).

[14] Kuncheva, L. I. and Whitaker, C. J., "Using diversity with three variants of boosting: Aggressive, conservative, and inverse," *Lecture Notes in Computer Science* , 81–90 (2002).

[15] Freund, Y. and Schapire, R., "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences* **55**, 119–139 (1997).

[16] Provos, N., "Defending against statistical steganalysis," in [*10th USENIX Security Symposium*], **10**, 323–336, Citeseer (2001).

[17] Westfeld, A., "F5-a steganographic algorithm: High capacity despite better steganalysis," in [*Information Hiding: 4th International Workshop, IH 2001, Pittsburgh, PA, USA*], (April 25-27 2001).

[18] Sallee, P., "Model-based steganography," *Lecture Notes in Computer Science* , 154–167 (2003).

[19] Fridrich, J., Goljan, M., and Soukal, D., "Perturbed quantization steganography," *Multimedia Systems* **11**(2), 98–107 (2005).

[20] Pevny, T. and Fridrich, J., "Merging Markov and DCT features for multi-class JPEG steganalysis," *Security, Steganography, and Watermarking of Multimedia Contents IX* , 1–13 (2007).

[21] Pevny, T. and Fridrich, J., "Merging Markov and DCT features for multi-class JPEG steganalysis," *Security, Steganography, and Watermarking of Multimedia Contents IX* , 1–13 (2007).

[22] Liu, Q., Sung, A., and Qiao, M., "Improved detection and evaluation for JPEG steganalysis," in [*Proceedings of the seventeen ACM international conference on Multimedia*], 873–876, ACM (2009).

Ensemble Systems for Steganalysis

S. Bayram, L. Li, A. E. Dirik, H. T. Sencar, N. Memon

***Abstract*—There are two key questions related to practical deployment of a steganalysis system: First, how to incorporate several steganalyzers together effectively, and second, how to limit the computational cost of such a system. To address these questions, we focus on the machine learning aspect of steganalyzer design and introduce two ensemble systems, namely, a hierarchical ensemble of classifiers based approach and a decision tree based approach. Both approaches provide systematic procedures to incorporate several steganalyzers together to improve detection performance. The ensemble of classifiers based approach significantly improves the scalability and flexibility of the steganalysis system. The decision tree based approach, alternatively, aims at minimizing the computational cost of steganalysis. To demonstrate the potential of both approaches, a series of experiments are performed on well known steganography and steganalysis techniques.**

I. Introduction

Multimedia steganography involves communicating by embedding information in seemingly innocuous media data. The objective of steganalysis then is to detect such steganographic communication and perhaps even to obtain definitive information on the steganographic technique being employed so that the hidden information can be extracted.

In recent years, there has been a large increase in research to develop better steganalysis techniques. As a result, both the design of steganalysis tools (steganalyzers) and their ability to accurately identify steganographic communication have witnessed significant improvements. Early steganalysis techniques were primarily developed by targeting specific steganographic techniques [1], [2], [3], [4]. These techniques perform very accurately when used against the specific steganographic embedding method they are designed to detect. Nevertheless, the approach implicitly assumes the knowledge of the steganography technique used for embedding. To remove this assumption, later steganalysis techniques adopted a more general approach by essentially focusing on the statistical properties of cover data (i.e., cover-objects used as the host for embedded information) and identifying those which are likely to exhibit significant change after the embedding operation.[1] This approach to steganalyzer design is often referred to as universal or blind steganalysis. Although universal steganalysis techniques perform less accurately than targeted methods, they can potentially identify a wide range of steganographic techniques [5], [6], [7], [8], [9], [10], [11].

In parallel to the development of increasingly sophisticated steganalysis techniques, a number of carefully designed steganographic techniques have begun to emerge. These embedding techniques are crafted to avoid detection by specific classes of steganalysis techniques [12], [13]. At the same time we are witnessing an increasing threat posed by steganography as the tool of choice used to exfiltrate sensitive data in corporate espionage and state sponsored espionage[14], [15], [16]. Therefore the next stage of research in steganalysis has led to the exploration of systems that combine many different steganalysis techniques. The argument here being that in a practical scenario, one does not know what specific technique to look for and the best chances of successful detection is achieved by deploying multiple steganalysis techniques as it would be difficult for the adversary to craft a steganographic embedding method that can escape detection from multiple steganalyzers each employing a different approach for detection.

Now, any system that is based on multiple steganalyzers should leverage the strengths of the individual techniques that it combines by delivering a superior performance then any single one of them. At the same time the system should be computationally affordable. Given the fact that most steganalysis techniques have at their core a machine learning component, these two objectives can be achieved by ensemble systems. Essentially, an ensemble system utilizes multiple machine learning classifiers in its construction to build a stronger overall classifier. Ensemble systems are known to perform better than single classifier systems in a wide variety of application scenarios [17], [18], [19].

In this paper, we introduce two ensemble systems for steganalysis. More specifically, we consider a hierarchical ensemble of classifiers based approach and a decision tree based approach. In addition to performance enhancements, the two systems offer a different set of advantages. Specifically, the hierarchical approach provides a modular and scalable design allowing incremental addition and removal of (steganalysis or steganography) techniques and incrementally incorporating additional training data. The decision tree approach on the other hand provides the ability to limit computational cost without significant reduction in performance.

The broad outline of the paper is as follows: In the following subsections we provide a review of related work in the field and provide a more formal problem definition. In Section II, we describe in detail the hierarchical ensemble of classifiers based construction. The decision tree based steganalysis system is introduced in Section III. Experimental results are reported in Section IV and conclusions are given in Section V.

A. Related Work

The literature in machine learning includes a vast amount of work that applies to information fusion and ensemble classifiers (also known as meta-classifiers). However, there are only a limited number of studies that apply these methodologies

[1]It should come as no surprise that many image steganalysis techniques subsequently found application in detecting digital image tampering.

to steganalyzer design. Research on steganalysis has focused primarily on designing accurate steganalysis techniques by identifying more discriminative features. There are only a few studies that have attempted to fuse information available from multiple (both universal and specific) steganalysis techniques so that the resulting steganalyzer is more effective against a wider variety of steganography techniques. For example, in [20], [21], to improve steganalysis accuracy Kharrazi et al. proposed a composite steganalyzer based on pre-classification and post-classification information fusion strategies. Similarly, Pevny et al. [22], [23] introduced a multi-class steganalysis technique that classifies a given stego-object to a known steganographic technique through combining binary classifiers under a one-vs-one strategy. Despite their promising results, neither of these approaches have fully utilized the body of knowledge on building ensemble systems.

Ensemble classifiers were in fact introduced to improve the performance of individual classifiers. An ensemble classifier is a system that make classification decisions based on the decisions from its base classifiers. The base classifier could be any simple and effective classifier that is easy to implement. Examples of popular base classifiers include linear discriminant, decision trees such as C4.5 [24] and Cart [25], etc.), or even Support Vector Machines (SVM) [26]. AdaBoost [27] and random forests [28] are two of the most widely used ensemble classifier systems given their theoretical foundation and practical success in many applications. AdaBoost focuses on the data instances which are initially incorrectly classified during training, and aims to build more specialized classifiers by iteratively putting more weight on these misclassified instances. By a weighted majority vote of the base classifiers generated in the boosting iterations, an AdaBoost classifier is able to achieve better accuracy than its base classifiers. With the random forest approach, each individual tree utilizes random samples from the original data and a unit vote is made because there is no discrimination of the base trees. The random forest is known to be more robust when the features or the labels are noisy.

In addition to accuracy, computational complexity of a classification system is also a practical concern in applied machine learning. One important aspect affecting computational complexity can be measured by the cost of retraining when new data is available and the classifier needs to be reconstructed. A cost-efficient machine learning system will have the property of incremental learning [29] if the retraining process only involves the new data. Another aspect relates to the computational cost of acquiring or extracting features, which corresponds to the notion of testing cost or feature acquisition cost in machine learning. A computationally efficient classifier should, on average, incur minimal computational cost for classifying a test instance. There are many approaches for optimizing the test cost of single classifiers [30], [31], while limited research ([32], [19]) has been done to optimize the testing cost of meta-classifiers.

B. Basic Framework

In this subsection we outline the basic steganalysis that we consider in the rest of the paper. Within this context we then give a brief outline of the two ensemble systems that we propose and evaluate.

In the rest of this paper we view a steganalyzer as essentially consisting of two main steps. The first step involves identifying a specific set of statistics (more commonly referred to as features) derived from cover-objects that are in general sensitive to steganographic embedding. These features describe some phenomena influenced by steganographic embedding and they often have high-dimensionality. The second step involves the use of machine learning algorithms to automatically generate classification models from extracted features that can discriminate between cover-objects and steganographically embedded objects, i.e., stego-objects. For practical purposes, these two steps can be isolated from each other and a steganalysis technique can be represented by its feature vector used for classification. Therefore, in the rest of the paper each steganalyzer will be identified by its feature vector $\mathbf{f}_j \in \Re^{d_j}$, $j = 1, 2, ..., N$, where N is the number of available steganalyzers and d_j is the dimensionality of the feature vector associated with steganalyzer j. To have better overall performance a steganalysis system can combine feature vectors from many steganalyzers.

From an operation point of view, a steganalysis system typically works in two phases: the offline phase and the online phase. In the offline phase, the system is initially built from scratch. This is realized through a process called training where feature data is used to train machine learning algorithms needed for classification. The feature data is obtained from a training set of objects with labels $Class_i$, $i = 1, 2, ..., M$, where each class is obtained by embedding random messages to a large set of cover-objects using different steganographic techniques and M is the total number of classes including the set of cover-objects. Training is a computationally intensive task but if it is performed one time, the complexity is surmountable. During the online phase, the system is tested against given stego- or cover-objects. The performance of a steganalyzer is measured as the average accuracy in differentiating objects belonging to different classes at varying message embedding rates. Several very comprehensive benchmarking studies have been performed to compare existing steganalysis techniques at an equal footing [21], [22], [33]. In this phase, computational resources are likely to be scarce and decision cost (testing cost) is the primary concern.

Given the above framework, we present two ensemble of classifiers based steganalysis system. The first systems uses a hierarchical ensemble of classifiers based approach and provides a procedure that utilizes feature vectors from many steganalyzers to build a multi-class classification system which can distinguish stego-objects created by different steganographic methods from each other and from cover-objects. A basic outline of the scheme is displayed in Fig. 1-a where each module serves as a binary classification system built by combining many boosted ensembles of classifiers. Essentially, the modules take as input all the features and each module specializes in only detecting one steganography technique. Decisions of the individual modules are later consolidated to make a final decision on the class a given object may belong. The resulting system not only improves detection performance but it also supports incremental addition of new steganalysis and steganography techniques and removal of existing ones by adding, removing and updating modules. Moreover, due its incremental learning capability, when new sets of stego- and cover-objects are available for training, the system is able avoid retraining with the previously used training data by expanding ensembles in each module.

The second system we present is based on a decision tree approach. This system aims at minimizing the computational cost of classification. To our knowledge, the design of steganalyzers that perform effectively with limited computational power has not been considered in the literature. Instead, feature selection has been usually employed as a pre-processing step to reduce the cost incurred by feature measurement. In practice, it is desirable that the computational cost associated with feature acquisition or measurement should be taken into account when designing cost efficient steganalysis systems. The decision tree based system attempts to achieve this while maintaining the classification accuracy and without performing feature selection. Fig. 1-b depicts a simplified diagram of the ensemble tree classifier. A base decision tree classifier and its boosted versions are organized as a meta-classifier. In the resulting decision tree meta-classifier, leaves represent classifications or decisions and internal nodes correspond to features. Starting at the root node, at each level of tree, value of a feature determines the path to be taken until a leaf is reached. Each path from root to leaves (like the one marked in red) is a decision rule and requires computation of a subset of features in sequence as opposed to computing of all features in advance.

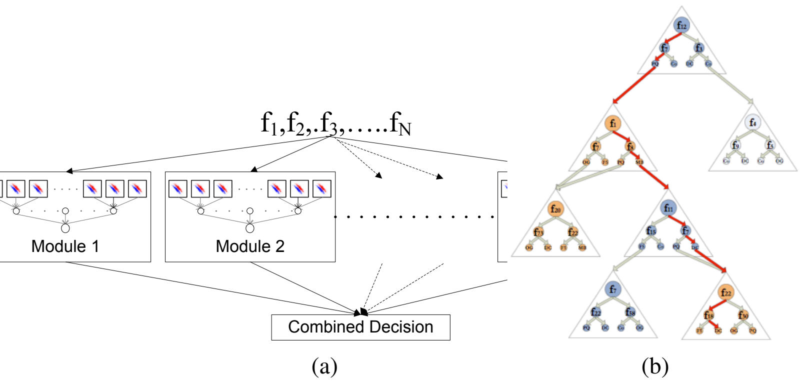


Fig. 1. Simplified structure of the two ensemble systems: (a) hierarchical ensemble of classifiers based approach and (b) decision tree based approach .

To evaluate the performances, both systems are built by combining feature sets associated with four universal steganalysis techniques, including wavelet based statistics (WBS) [7], binary similarity measures (BSM) [5], merged Markov and DCT features (MRG) [34], and joint density features (JD) [35], and tested against four steganography techniques that include Outguess [36], F5 [37], model-based (MB) [38] and perturbed quantization (PQ) based steganographic embedding techniques [12]. In the following sections, both approaches are explained in more detail.

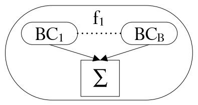


Fig. 2. The first level of HEC. In this level, for each feature set, several base classifiers are ensembled through boosting to improve the result of individual steganalyzers.

II. Steganalysis Using a Hierarchical Ensemble of Classifiers

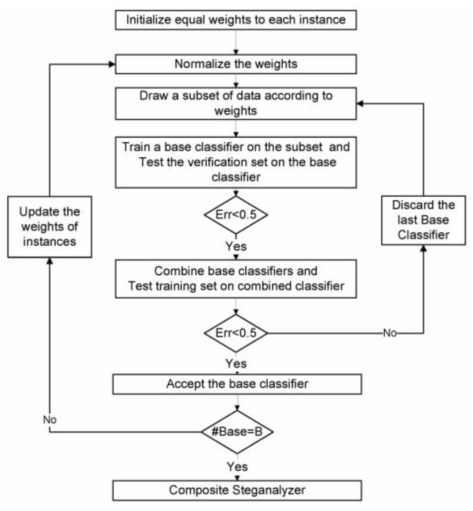
In this section we introduce a Hierarchical Ensemble of Classifiers (HEC) based steganalysis system which is a multi-class classification system that draws from the concept of boosting and provides a systematic procedure to combine many steganalysis techniques to yield more accurate decisions. A key property of the system as explained below is its incremental updatability and incremental learning property.

A. The Structure of HEC

In the design of a HEC based system, each steganography techniques (as well as the set of cover-objects) that need to be identified by the steganalysis system is associated with a class. Each class is built as a binary classifier which utilizes all the features vectors and makes a decision as to whether a given object belongs to that particular class or not. Boosting is used to combine many simple base classifiers into ensembles, and a binary classifier combines many ensembles, as many as the number of steganalysis technique we want to merge together. To be more clear, in HEC instead of building one multiple-class classifier, multiple binary classifiers are built. In this structure there are 3-levels. In the first level (see figure 2), boosting is used to combine many simple base classifiers to improve the performance of each steganalysis technique. In the second level of the hierarchy, the decision of all boosted steganalyzers are ensembled to get a combined decision of all steganalyzer techniques that are being used. These two levels are utilized to build a binary classifier for each class and in the last level of the HEC, the final decision is made according to the decisions of all binary classifiers. Below we describe the necessary steps required to build a classification module for each class. The same steps need to be repeated for all classes. Individual decisions from all the binary classifiers are then combined to make a final decision.

1) Improving Accuracy by Boosting: As mentioned earlier, in the first level of HEC, the aim is to improve the performance of each universal steganalysis technique by building ensemble classifiers through boosting (Figure 2). For this purpose, an AdaBoost-like iterative algorithm whose steps are shown in Figure 3.

This algorithm takes as input a training data set, a validation data set, an integer B which defines the number of iterations, and a base classifier BC which can yield probabilistic values rather than binary decisions. The training and validation data consist of m and l instances, respectively, where $x_i \in \Re^{d_n}$ represent features associated with steganalyzer n extracted from training instance i, and $y_i \in \{-1, +1\}$ is a label showing whether or not instance i belongs to the particular class of



Initialize equal weights to each instance
Normalize the weights
Draw a subset of data according to weights
Train a base classifier on the subset and Test the verification set on the base classifier
Err<0.5
Yes
Update the weights of instances
Discard the last Base Classifier
Combine base classifiers and Test training set on combined classifier
Err<0.5
No
Yes
Accept the base classifier
No
#Base=B
Yes
Composite Steganalyzer

Fig. 3. Boosting algorithm used for generating an ensemble classifier for each feature vector.

objects for which the binary classifier is being built. Let us refer to one of the classes as positive and the other class as negative according to their labels y_i.

In the first step of the algorithm, equal weights are assigned to each of the training data instances, i.e., $w_0(i) = 1/m$, and after each iteration of the algorithm, these weights are updated and normalized to form a probability distribution as

$$D_t(i) = \frac{w_t(i)}{\sum_{i=1}^{m} w_t(i)}, \tag{1}$$

where D_t represent the distribution of weights assigned to each instance of the training data after tth iteration. In each iteration, a random subset from the training data is drawn according to D_t and a hypothesis h_t is generated on this random subset using the base classifier BC. The error for the current base classifier BC_t is computed on the validation data set as

$$\epsilon_t = \frac{1}{l}\sum_{i=1}^{l}[h_t(x_i) \neq y_i]. \tag{2}$$

It should be noted that, in contrast to AdaBoost where error is defined as the sum of the weights of misclassified training instances, the error here is defined as the proportion of the misclassified instances in validation data to get a more realistic measurement. If the resulting error is greater than 0.5, newly trained base classifier is discarded and a new subset is drawn. Otherwise, if $\epsilon_t < 0.5$, the base classifier is accepted and its error is normalized as

$$\beta_t = \frac{\epsilon_t}{(1-\epsilon_t)}. \tag{3}$$

During each iteration, the accepted base classifier is incorporated into the ensemble using a weighted sum rule to obtain the composite hypothesis H_t. According to the weighted sum rule, the probability that an instance belongs to the positive class can be defined as:

$$P_t(i) = \frac{\sum_{i=1}^{t} p_t(i).log(\frac{1}{\beta_i})}{\sum_{i=1}^{t} p_t(i).log(\frac{1}{\beta_i}) + \sum_{i=1}^{t} n_t(i).log(\frac{1}{\beta_i})} \tag{4}$$

where p_t is the probability the instance is classified as positive by the base classifier (BC_t) and $n_t = 1 - p_t$ is the probability the instance is classified as belonging to the negative class. Similarly, the probability that an instance is classified as negative by the ensemble can be defined as $N_t = 1 - P_t$. Therefore a positive instance is accepted to be classified correctly if $P_t > N_t$ and vice versa if $N_t > P_t$. As can be seen in these equations, base classifiers are awarded higher weights if their error is smaller. The error of the ensemble can be computed by testing the composite hypothesis, $H_t = [P_t > N_t]$, on the training data as

$$E_t = \sum_{i=1}^{m} D_t(i).[H_t(x_i) \neq y_i]. \tag{5}$$

If $E_t > 0.5$, then the base classifier which was trained last is again discarded and a new data set is drawn; otherwise, the computed error is normalized and new weights are calculated as

$$\varrho_t = \frac{E_t}{(1-E_t)}, \tag{6}$$

$$w_{t+1}(i) = w_t(i).\varrho_t^{1-[H_t(x_i)\neq y_i]}. \tag{7}$$

It should be noted that in this step validation data cannot be used to compute the error since we want to update the weights of the training data. One other difference of this boosting scheme from AdaBoost is that in AdaBoost when weights are being updated, only the hypothesis h_t generated by the newest base classifier is considered; whereas in HEC, we use the hypothesis H_t generated by the whole ensemble. This is because we want to focus on instances that are classified incorrectly by the whole ensemble rather than the ones classified incorrectly by the last base classifier only.

These steps are repeated for all iterations. Finally, after Bth iteration the error of the ensemble can be computed and normalized on the validation data, respectively, as

$$Es = \frac{1}{l}\sum_{i=1}^{l}[H_B(x_i) \neq y_i], \tag{8}$$

$$\gamma_{j,k} = \frac{E_s}{(1-E_s)}. \tag{9}$$

It is clear that E_s cannot be higher than 0.5 since at earlier steps it is ensured that all base-classifiers and the ensemble classifier have an error probability less than 0.5. At the end of this procedure an ensemble classifier using features associated with one steganalysis technique is generated. It has to be repeated for every steganalyzer in each class, and corresponding γ_{j_k} values need to be obtained for $j = 1, 2, .., N$, $k = 1, 2, ...M$.

2) Combining the Decisions of Ensembles: The next step involves incorporating decisions of many boosted classifiers built in the previous step so that a final decision as to whether or not a given object belongs to a particular class can be made. In [39], Kittler argued that the weighted sum rule is preferable when combining weak classifiers together and the weighted product rule is preferable for strong classifiers. Since ensemble classifiers obtained up and until this step are already

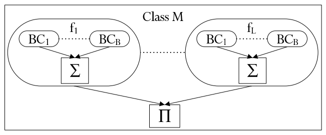


Fig. 4. Combination of boosted steganalyzers to make an overall decision as to whether an object belongs to $Class\ M$.

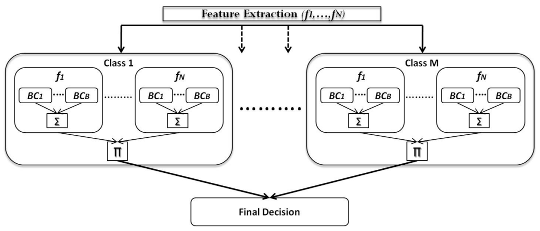


Fig. 5. Overall hierarchical ensemble based classifier

strengthened by boosting, we decided to ensemble their decisions by the weighted product rule as displayed in Figure 4. Here, the weights are determined as the inverse of boosted classifier's overall error. Correspondingly, the probability of an instance to belong to the class in question, ϱ_{class}, is computed as following

$$\varrho_{class} = \frac{\prod_{i=1}^{N} P_B(i).log(\frac{1}{\gamma_i})}{\prod_{i=1}^{N} P_B(i).log(\frac{1}{\gamma_i}) + \prod_{i=1}^{N} N_B(i).log(\frac{1}{\gamma_i})} \tag{10}$$

In Eq. 10, essentially, the weights determine how discriminative the features associated with a steganalysis technique is in detecting objects belonging to a particular class, and this idea lies at the core of HEC based approach.

3) Testing: During testing, the object in question first undergoes feature extraction. Computed features are then classified by the base-classifiers comprising each ensemble classifier and the resulting decisions are combined by the weighted sum rule using weights computed in the off-line training phase. Decisions of each ensemble are then combined by computing their weighted product. This results in M scores being obtained at the output of each classification module associated with the M object classes. Finally, the object is attributed with the class that yields the highest score. The overall HEC based steganalysis system is displayed in Figure 5.

B. Advantages of the Proposed Composite Steganalyzer

The resulting steganalysis system has the following main advantages. First of all due to the underlying boosting methodology it ensures that for each steganalysis technique, a strong classifier with near-optimal error performance is constructed [40]. Second, the resulting classification system organizes feature vectors associated with each steganalysis technique with respect to their discrimination power in each class. Therefore, while some steganalyzers are awarded highly in discriminating some stegangoraphy methods, they might be awarded less for others. Another advantage of HEC based system is that it can be naturally extended to multi-class steganalysis scenarios where not only discrimination of stego- and cover-objects is an issue, but also the steganographic technique used in generation of the stego-object is in question.

The HEC based system is modular. When a new steganalysis technique has to be added to the system, this is realized by training and adding an ensemble classifier to each class in the system. Similarly, adding a new class, i.e., steganography technique, requires training a binary classification system composed of many ensembles. Removal of a steganalysis technique or a class, however, is performed by eliminating the corresponding ensembles from all classes or the class itself completely.

Another inherent advantage of this structure is its ability to allow incremental changes. When a new batch of a data becomes available for further training, system can be updated by adding new base-classifiers to each ensemble and running the boosting scheme with the new data only. Since existing base classifiers are already trained using earlier data and the weights are computed accordingly, these weights can be considered to act as a memory that broadly summarizes the old data and, therefore, retraining of the entire system will not be needed.

III. Decision Tree Based Steganalysis Approach

The approach described in this section applies a meta-classifier CoCoST [19] based on decision trees to combine many steganalysis techniques together. Decision trees (DT) are widely used in machine learning, and there are two reasons to utilize a decision trees in steganalysis. First, decision tree classifiers have accurate predictive capabilities and are easy to train. The second reason is that the structure-wise prediction process in a decision tree is inherently incremental, and it is computationally very efficient. These two properties are very important in steganalysis as they address two of the three essential practical concerns in deploying steganalyzers, namely accuracy and speed.

In a decision tree, as long as the path for a given data instance does not include all the features, the number of features needed to label this example is reduced. Hence, for decision trees to make predictions, we do not need to provide all the feature values beforehand. Instead, we can extract feature values when the encountered decision tree node requires a certain feature value. This is important because it not only exempts us from pre-computing all the feature values for each steganalyzer, but also reduces the average number of features needed to detect a stego-object.

These characteristics of decision trees are very desirable when there is cost associated with acquiring each feature. The expected cost for making a decision for an unknown instances random data is defined as the expected cost of a decision tree. With uniform cost, the expected cost becomes the average number of features used for predicting each instance. To optimize the expected number of features needed for predicting each instance, different classification trees are proposed in [31] [30]. However, individual trees are not accurate enough, and we therefore use a meta-classifier which consists of new cost-efficient trees called the LASC (Looking Ahead Suppressed Cost) tree.

In an application scenario where some features are not available or more expensive to extract, the decision tree based approach addresses the problem of achieving good accuracy while using as few features as possible. The advantage for using the LASC individual tree classifier and the CoCoST meta-classifier is that both the base classifier and the meta-classifier are built cost-efficiently. Also using the proposed classifier, the sensitivity to cost is flexible when the node size is different. Taking size information into account when designing tree classifier yields a more cost efficient tree in practice [19]. Next, we present the structure of CoCoST based steganalysis system and its advantages.

A. The structure of the CoCoST classifier

1) LASC - The Base Tree Classifier: Optimal decision trees have been proven to be NP-hard to build [41], and hence various heuristics have been developed for this purpose. The most widely used decision tree is the C4.5 by Quinlan [24]. The heuristic of the C4.5 tree is the normalized entropy gain for choosing a feature to split. And its cost efficient variants [31][30] take the form of the entropy gain over the cost associated with acquiring that feature value. To address the computational concerns of steganalysis, we use the LASC decision tree algorithm as the base classifier. The LASC tree is more cost efficient and suitable for steganalysis as it also considers the cost of acquiring feature values.

There are two key characteristics of the LASC algorithm. The first one is that LASC utilizes not only the entropy gain, but also the size of the node where the size of the node is the proportion of training instances in each node. For example, the root node has a size of one, while the leaf nodes have much smaller sizes. None of the cost-efficient variants of C4.5 trees take this information into consideration. However, at the root node, where the node size is big and every instance gets tested against the root feature, we want the root feature to be very cheap and efficient. On the contrary, at the leaf nodes where few instances enter, we can tolerate using more number of features and features with expensive cost for a better discrimination power. The LASC heuristic is expressed as

$$H = \frac{\triangle I}{freq^{\alpha} C + (1 - freq^{\alpha})}. \tag{11}$$

In Eq. 11, $\triangle I$ is the 'look ahead' entropy gain which represents the information we obtain by choosing this feature to separate the data; C is the cost associated with each feature which is assumed to be uniform over all features; $freq$ denotes the size of the node which is the fraction of training instances at that node; and the constant parameter α designates the sensitivity of the heuristic to cost.

The core idea behind the LASC heuristic is that the sensitivity to cost is suppressed as the tree grows and the size of the node gets smaller. Consider the root node whose size is one. The root node is a big node and each instance has to be tested against the root feature which is to say the root feature has to be extracted from all instances. Then, $1-freq^{\alpha}$ will be around zero and the term $freq^{\alpha}C$ will dominate. Therefore, the heuristic will choose the most cost-efficient feature. As the tree is expanded and the nodes become smaller, fewer data instances will enter the lower level nodes at which we may deploy more discriminative features. In this case, the term $freq^{\alpha}$ gets smaller and the term $1-freq^{\alpha}$ dominates. Correspondingly, the heuristic now picks the feature that provides the most information. In summary, a LASC heuristic yields an efficient decision tree by reducing the average number of features used for classification.

The other key characteristic of the LASC tree is that when calculating the entropy gain, it allows looking ahead one or more steps (k steps) by looking for a better combination of features instead of looking for individually good features when calculating the heuristic and building the tree. It should be noted that there is a trade-off between building better trees and the training time. Although looking more steps ahead will help build a more accurate decision tree, this will lead to longer training times. Therefore, in this paper, we use $k = 1$ looking ahead which takes $O(n^3)$ as training cost. The LASC tree is constructed recursively by selecting the most fitting feature according to the heuristic score.

2) Inverse Boosting to Diversify the Pool: The intuition for boosting is to generate more diversified and independent classifiers. As opposed to the previous system described in Section II, which deploys an adapted version of AdaBoost, the CoCoST classifier utilizes standard AdaBoost [27] as well as inverse boosted decision trees as its base classifiers.

Inverse boosting was originally introduced to create ensembles of classifiers with high varieties [42], [43]. Instead of raising the weight of misclassified instances, we raise the weight of correctly classified instances, and the updating function for the new weight is the same as standard boosting but with the sign flipped for the indicator. Given the weights $D_t(i), i = 1, 2, ..., M$ and the error rate ϵ_t of the current classifier, updated weights for each inverse boosting round are obtained as

$$D_{t+1}(i) = \frac{D_t(i) exp(\beta_i I(y_t(i) = h_t(\vec{X}_i))}{Z_t}, \tag{12}$$

where Z_t denotes the normalizing vector, I is the indicator function which raises the weight when the instance is misclassified and reduces it when the instance is correctly classified, i.e,

$$I(y_t(i) = h_t(\vec{X}_i)) = \begin{cases} 1, & \text{if } y_t(i) = h_t(\vec{X}_i), \\ -1, & \text{otherwise;} \end{cases} \tag{13}$$

and $-\beta$ is a parameter calculated from the error rate ϵ as

$$\beta_t = \frac{1}{2} log \frac{1-\epsilon_t}{\epsilon_t}. \tag{14}$$

By performing boosting and inverse boosting on the LASC tree classifier, we obtain a more diversified pool of classification trees. It should be noted again that standard boosted trees tend to focus on instances that are hard to classify, while inverse boosted trees are cheaper and tend to focus on easy instances using fewer features. It is worth mentioning that depending on what speed we want the boosting performance to converge, the weights can be updated more aggressively or conservatively as compared with Eq. 12.

3) Combining the Trees by Stacked Generalization: The pool of classification trees obtained from the LASC tree by boosting and inverse boosting at difference sensitivities, α, are incorporated together to obtain the CoCoST classification system. This is realized by using a method called stacked generalization. In stacked generalization, each boosted tree predicts on a validation data set and their decisions are treated as the new feature set to build a meta-classifier.

The detailed process for generating the CoCoST meta-classifier is as follows. Given a pool of diversified individual decision trees, either boosted or inverse boosted, we apply these trees on the validation data set. These boosted/inverse-boosted candidate trees have different sensitivity to computational cost, and they also exhibit accuracy biases on different stego-objects. Given the validation dataset, we treat the detection result of all these trees as feature values, and the meta-classifier is trained on the decisions validation data, where rows denote stego-objects and the columns denote the decision results of each candidate tree classifier. The final meta-classifier of CoCoST, is a big tree classifier where each internal node is actually a boosted LASC tree.

B. Advantages of the Decision Tree Based Approach

The decision tree based steganalysis system offers significant computational advantages. It effectively addresses both performance and computational complexity concerns. During online testing phase, the system will ask for feature values incrementally until a decision can be made as to which class the instance belongs to. Therefore, there is no need to pre-compute any features. The design ensures that the average computational cost for detection is minimized while still ensuring accurate detection. This system also possesses the ability of incremental learning. When new training data is available, it requires generating a new set of trees using the fresh data and retraining the meta-classifier. However, addition or removal of steganalysis techniques will trigger retraining of the whole system.

IV. Experiments and Results

In this section we present an experimental evaluation of the two systems we have proposed. First, we introduce the universal steganalysis techniques and steganographic methods used in the experiments. Then, we describe our experiments and report their results.

A. Studied Techniques

To test the performance of the proposed steganalysis techniques both systems were built using well known steganography and steganalysis techniques. In general, a steganography technique can be characterized by how it attempts to achieve undetectability by a steganalysis technique. Below we list four JPEG based steganography techniques and briefly state their basic approach to limiting the embedding distortion they induce in a cover-object. All these techniques embed data in DCT domain without any noticeable visual artifacts.

- *Outguess* [36] tries to preserve DCT coefficient histograms by selecting DCT coefficients that yield least embedding distortion. Outguess re-compresses cover-images prior to embedding at JPEG quality factor 75.
- *F5* [37] embeds by decrementing absolute values of randomly chosen DCT coefficients by one. It employs the notion of 'matrix embedding' to minimize the number of changes made to DCT coefficients. Similar to Outguess, F5 re-compresses cover-images at JPEG quality factor 80.
- *Model Based (MB) Steganography* [38] uses a parametric model of DCT coefficient distributions, obtains model parameters, and then performs embedding in a way that preserves those models. It does not perform re-compression on cover-images.
- *Perturbed Quantization (PQ) Steganogprahy* [12] is based on an information reduction operation like re-compression. During compression, quantization for a certain subset of DCT coefficients is slightly perturbed to embed message bits. The perturbation is by modifying the coefficient so that it quantizes to a designated quantization level. Since stego-images generated by PQ technique very closely mimic double compressed images, it is significantly less detectable than other techniques.

Steganalysis system utilizes features introduced by the following steganalysis techniques.

- *Binary similarity measures (BSM)* [5] refers to features extracted in spatial domain. The intuition behind the steganalysis technique is that the embedding operation distorts the correlation between the bit planes of an image. The change in bit plane correlations are detected through 18 features extracted from seventh and eighth bit planes of cover- and stego-images.
- *Wavelet based steganalysis (WBS)* [7] extracts features in wavelet transform domain. To detect steganographic embedding, WBS aims at capturing underlying statistics of wavelet sub-band coefficients. For this purpose, 72 features are extracted from transformed intensity of the color channels. These 72 features comprise first order statistics (mean, variance, skewness, kurtosis) of sub-band coefficients and higher order statistics computed from linear prediction error.
- *Merged DCT and Markov features* are obtained by combining two sets of features. In [34], authors proposed the use of a process called calibration to detect stego-images. Calibration is used to estimate macroscopic properties of the cover-image from the stego-image. Then, 23 spatial and DCT domain features are used to capture the difference between an image and its calibrated version. DCT features are then obtained by extending these features to a larger set of 193 features [34] . In a similar manner, [44] modeled the differences between absolute values of neighboring DCT coefficients as a Markov process and computed 81 features from resulting probability transition matrices. To improve the steganalysis accuracy, [34] merged the 193 extended DCT features with the Markov features which resulted with 274 calibrated features. The steganalysis technique based on merged features is

reported to have significantly better detection accuracy than other universal steganalysis techniques.

- *Joint Density features* [35] are proposed to capture traces of embedding in terms of relations between DCT coefficients of neighboring blocks. Features are obtained from joint distribution of inter and intra block DCT coefficients. Here we solely utilize 169 intra block joint density DCT features due to their superior detection capabilities.

B. Dataset

For the experiments we used publicly available Greenspun's image set which contains 1800 images taken under varying conditions. Prior to steganographic embedding, black borders around the images are removed and all images are converted into grayscale and re-saved in JPEG format with quality set to 100. Out of all the images, the ones with significantly low embedding capacity are discarded, leaving us with a total of 1600 images. Then four classes of stego-images were created by embedding all the 1600 cover images with each of the four steganography methods separately.

For embedding, random binary strings are generated as messages. Message lengths are determined adaptively for all images depending on the number of non-zero DCT (NZ-DCT) coefficients in the image. For example, 0.10 BPNZ-DCT (bits per non zero DCT coefficient) embedding rate corresponds to a message length in bits that is equal to 10 % of NZ-DCT coefficients of the cover-image. The class of cover-images were then subjected to double compression operation to create another class of objects. Hence, a total of six classes of images, namely, cover image set , double compressed cover image set, and four sets created from cover image set using Outguess, F5, MB and PQ steganography techniques, were generated. Then the two steganalysis systems proposed in this paper were tested against these classes to determine their discrimination accuracy.

C. Experimental Results

We conducted several experiments to determine the effectiveness of the two steganalysis approaches. The goal is to distinguish how well the two systems are able to identify the class of an unlabeled images from among six possible classes. When building the two systems we used 500 images for training, 500 for validation, and remaining 600 for testing.

1) Results with Multi-Class SVM: For the first experiment, we have concatenated all the feature vectors and obtain a single feature vector for each instance. We then designed a 6-class SVM classifier using 500 training and 500 validation data. For our experiments, we used publicly available libsvm package [45]. A grid-search is employed to find the best parameters for SVM classifier. The results of the experiment is presented in Table-I. According to these results, the overall accuracy is 82.61% however, it must be noted that the errors are primarily due to confusion between DC cover-image class and PQ stego-image class (49.25%) . This is not surprising given the fact that PQ embedding operation is deliberately designed to mimic double compression operation.

TABLE I
CONFUSION MATRIX OF SVM CLASSIFIER WITH GRID SEARCH

	Cover	Double Cover	F5	Outguess	MB	PQ
Cover	600	0	0	0	0	0
DC Cover	0	145	0	0	0	455
F5	0	0	600	0	0	0
Outguess	0	0	0	600	0	0
MB	39	0	0	0	561	0
PQ	0	154	0	0	0	446

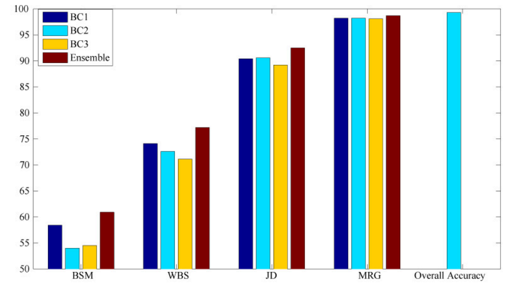


Fig. 6. Performance of boosted ensembles with SVM as the base-classifier in discriminating non-compressed cover class from other classes combined. Each cluster shows the impact of boosting on different steganalyzers. Within each cluster first three column represents the performance of the base classifiers designed in each iteration and the last column represents the performance of the ensemble.

2) Results on HEC Based Steganalyzer: In constructing HEC, we used SVM as the base classifier. We first conducted an experiment to show the performance improvement due to boosting based ensemble classifier, as depicted in Figure 2. The ensemble classifier is generated using up to three base-classifiers, $B = 3$. We designed a two-class classification scenario where the first class consisted of non-compressed cover images and the second class consisted of the pool of images from all four steganography techniques and from double compressed images. (It must be noted that when a single base-classifier is used the ensemble classifier reduces to an SVM classifier.) When building the HEC based systems features from all four steganalysis techniques (i.e., BSM features, WBS features, merged DCT and Markov features, and joint density features) are used individually and collectively to train the system. In a similar experiment, the same system is retrained this time to test its ability to differentiate F5 class from the other five classes combined. The results for the two experiments are presented, respectively in Figures 6 and 7. In both figures it can be seen that, although SVM is considered a strong classifier, the improvement due to ensemble classifier is considerable in most cases. This shows that through boosting even SVM classifier can be made stronger.

In the second set of experiments, we tested how the binary classification system given in Figure 4 improves the results by combining the decisions of different steganalyzer, i.e., ensembles. For this purpose, we designed four separate technique-specific classifiers to discriminate four classes (generated using PQ, ModelBased, OutGuess, F5 steganography techniques) from the class of cover and double-compressed cover images. (When designing a classifier for PQ steganography technique we used double-compressed cover image class and for the rest

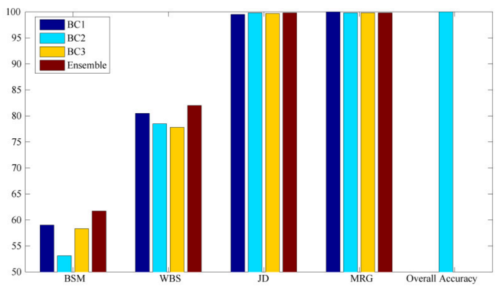


Fig. 7. Performance of boosted ensembles with SVM as the base-classifier in discriminating F5 class from other classes combined. Each cluster shows the impact of boosting on different steganalyzers. Within each cluster first three column represents the performance of the base classifiers designed in each iteration and the last column represents the performance of the ensemble.

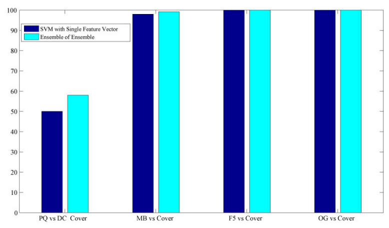


Fig. 8. Comparison of single SVM (first column) and HEC (second column) performances under four technique-specific steganalysis scenarios.

non-compressed cover images are used.) Using HEC based approach, a new classifier is designed for each technique. Also, for comparison, feature vectors associated with four steganalyzers are combined into a single feature vector and an SVM classifier is built for each technique. Results corresponding to the four test scenarios are given in Figure 8 where first column shows the result for SVM and second for HEC. From this figure, it can be seen that for F5 and Outguess both SVM classifier with combined feature vector and HEC provide perfect results and for Model Based steganography HEC is only marginally better. However, PQ steganography was completely indistinguishable from double compressed cover class with SVM classifier. HEC, on the other hand, improved the results by 7%. To the best of our knowledge, this is the best reported result against PQ steganography using these features. Although in [46] authors reported a performance of 70%, that result does not apply to all types of datasets such as the one used in this paper.

Finally, ability of the HEC based system to differentiate 6 classes of images from each other, a total of 3600 images, is summarized by the confusion matrix given in Table II. In this experiment, we tried different number of base classifiers $B = 3, 5, 7$ and found out that after $B = 5$ the results didn't improve. Overall steganalysis accuracy is computed to be 85.22%; however, it must be noted that the errors are primarily due to confusion between DC cover-image class and PQ stego-image class which yielded to an accuracy of 56.5%.

3) Results on Decision Tree Based Steganalyzer: Using the 500 training images for each class, we built a LASC tree and then applied boosting and inverse boosting to generate multiple decision trees. The steganalysis accuracy of a single LASC tree, CoCoST with 5 trees and CoCoST with 11 trees in discriminating one class of images from all other classes are shown in the bar graph given in Figure 9. It can be seen that except for DC Cover and PQ steganography classes, increasing the number of trees quickly saturates the accuracy to almost 95%. However, for the two classes increasing the number of boosted trees further do not yield an improvement in classification as expected.

The confusion matrix for 3600 test instances are shown in the Table III. The total accuracy for all the 6 classes are 86.92%, and the expected number of features used for each instance is 211.56 features, which is 60% less than the features that need to be acquired for each instance in the conventional steganalysis. Also, we see that except for DC Cover and PQ classes which we can only differentiate with an accuracy of 62.92%, the other classes are classified with an accuracy close to 95%. The reason for inferior performance in distinguishing DC Cover and PQ steganography classes is essentially due to insufficiency of the available features. Even by adding more boosted trees which are naturally more accurate on these two classes, accuracy does not increase much. In contrast, the detection accuracy for other classes increase significantly with more boosted trees.

The CoCoST classifier achieves an accuracy of around 88%

TABLE II
CONFUSION MATRIX FOR HEC BASED STEGANALYSIS SYSTEM USING SVM AS THE BASE-CLASSIFIER

	Cover	DC Cover	F5	Outguess	MB	PQ
Cover	600	0	0	0	0	0
DC Cover	0	332	0	0	0	268
F5	0	0	600	0	0	0
Outguess	0	0	0	600	0	0
MB	10	0	0	1	589	0
PQ	0	274	0	0	0	326

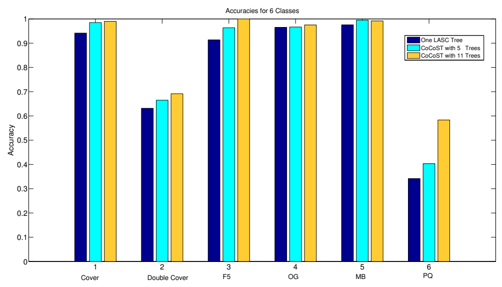


Fig. 9. Accuracies for the 6 Classes Using CoCoST Classifier with Different Number of Trees

TABLE III
CONFUSION MATRIX OF COCOST WITH 11 TREES

	Cover	Double Cover	F5	Outguess	MB	PQ
Cover	594	3	0	0	0	3
DC Cover	0	415	8	6	0	179
F5	0	0	600	0	0	0
Outguess	7	0	4	585	0	4
MB	2	0	0	0	595	3
PQ	25	219	3	2	1	350

while it significantly reduced the average number of features used for each data instance by 60%. In a scenario, where we want quick prediction and prefer to extract as few features as possible, the cost-efficient CoCoST classifier is a good choice. However, if our computing power can bear acquiring all the feature values and we want to get the best accuracy out of all the features, we may want to use a more complicated classifier, for example SVM, to get an even better accuracy. The reason that decision tree based multi-class predictor works slightly better than the purely boosted hierarchical classifier have two aspects. First, by introducing inverse boosting combined with standard boosting, prediction noise is suppressed by multiple diversified decisions from the pool of trees. Secondly, the CoCoST classifier is a meta-classifier which implies two training phases: one for the base classifier and the other for the meta-classifier. Therefore a stronger hypothesis and more complicated system is built and better accuracy can be reasonably expected.

V. Discussion and Conclusion

In this paper, we look at the steganalysis problem from a practical deployment point of view. Steganalysis research is mainly motivated to investigate the possible features that yields highest accuracy in identifying cover-objects subjected to steganographic embedding. Although design of better steganalysis techniques is a crucial goal, given the diversity of steganography techniques and their ever increasing sophistication, it is not realistic to assume a single technique will outperform others in identifying all types of steganographic techniques. Therefore in practical application scenarios, these competing techniques have to be incorporated together. From this perspective, the most important goal of a steganalysis system, that combines many individual steganalysis techniques, is to improve performance. To achieve this goal, in this paper, two ensemble based classification systems with distinct advantages are introduced.

It is shown through experiments that both systems improve (binary and multi-class) steganalysis performance as compared to conventional feature level fusion, where a single classifier is trained on joint features formed by stacking all the available features. The hierarchical ensemble classifier (HEC) based system mainly provides a modular and scalable structure as it allows for easy addition and removal of techniques and provides incremental learning ability such that when a new batch of data is available, new base classifiers can be trained and added to existing ensembles. However, HEC lacks on the computational efficiency aspect since for each test instance all the features have to be calculated and tested on a large number of classifiers.

Alternatively, the decision tree based system provides the ability to limit the computational cost in the online phase of steganalysis. to maintain a comparable accuracy to best classifiers such as SVM. The decision tree based steganalyzer is an online algorithm which requires values of new features dependent on previously provided feature values. Therefore, on average, only a few of all the features have to be computed before making a decision. The steganalyzer also exhibits characteristics of incremental learning as it is convenient to add new training data to the system. Limitations of the decision tree based classifier is that when new techniques are added or removed, the system has to be retrained.

VI. Acknowledgement

This material is based upon work supported by the Air Force Office of Scientific Research (AFOSR) under grant No FA9550-09-1-0087.

References

[1] Sorina Dumitrescu, Xiaolin Wu, and Zhe Wang, "Detection of lsb steganography via sample pair analysis," in *Information Hiding*, Fabien A. P. Petitcolas, Ed. 2002, vol. 2578 of *Lecture Notes in Computer Science*, pp. 355–372, Springer.

[2] Jessica J. Fridrich, Miroslav Goljan, Dorin Hogea, and David Soukal, "Quantitative steganalysis of digital images: estimating the secret message length," *Multimedia Syst.*, vol. 9, no. 3, pp. 288–302, 2003.

[3] Andrew D. Ker, "Steganalysis of lsb matching in grayscale images," *IEEE Signal Processing Letters*, pp. 441–444, 2005.

[4] J. Fridrich, D. Soukal, and M. Goljan, "Maximum likelihood estimation of secret message length embedded using pmk steganography in spatial domain," *In Proc. SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VII*, vol. 5681, pp. 596–606, 2005.

[5] I. Avcibas, M. Kharrazi, N. Memon, and B. Sankur, "Image steganalysis with binary similarity measures," *EURASIP Journal on Applied Signal Processing*, vol. 2005, no. 17, pp. 2749–2757, 2005, doi:10.1155/ASP.2005.2749.

[6] Ismail Avcibas, Nasir D. Memon, and Bülent Sankur, "Steganalysis using image quality metrics.," *IEEE Transactions on Image Processing*, vol. 12, no. 2, pp. 221–229, 2003.

[7] S. Lyu and H. Farid, "Steganalysis using higher-order image statistics," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 1, pp. 111–119, 2006.

[8] Jessica J. Fridrich, "Feature-based steganalysis for jpeg images and its implications for future design of steganographic schemes," in *Information Hiding*, Jessica J. Fridrich, Ed. 2004, vol. 3200 of *Lecture Notes in Computer Science*, pp. 67–81, Springer.

[9] Tao Zhang, Wenxiang Li, Yan Zhang, Ergong Zheng, and Xijian Ping, "Steganalysis of lsb matching based on statistical modeling of pixel difference distributions," *Inf. Sci.*, vol. 180, pp. 4685–4694, December 2010.

[10] Robert Buml, Johannes B. Huber, and Andr Kaup, "Steganographic system based on higherorder statistics," in *in Proc. SPIE Vol. 5020, Security and Watermarking of Multimedia Contents V*, 2003, pp. 156–166.

[11] S. Lyu and H. Farid, "Steganalysis using color wavelet statistics and one-class support vector machines," in *SPIE Symposium on Electronic Imaging*, San Jose, CA, 2004.

[12] J. Fridrich, M. Goljan, and D. Soukal, "Perturbed quantization steganography," *Multimedia Systems*, vol. 11, no. 2, pp. 98–107, 2005.

[13] Kaushal Solanki, Anindya Sarkar, and B. S. Manjunath, "Yass: Yet another steganographic scheme that resists blind steganalysis," in *9th International Workshop on Information Hiding*, Jun 2007.

[14] P. T. Anitha, M. Rajaram, and S. N. Sivanandham, "Analysis of detecting steganography contents in corporate emails," *International Journal of Research and Reviews in Electrical and Computer Engineering (IJRRECE)*, vol. 1, no. 2, June 2011.

[15] Sally Adee, "Spy vs. spy," *IEEE Spectrum Magazine*, August 2008.

[16] Sally Adee, "Russian spies thwarted by old technology?," *Discovery News*, June 2010.

[17] Yan-Shi Dong and Ke-Song Han, "Boosting svm classifiers by ensemble," in *Special interest tracks and posters of the 14th international conference on World Wide Web*, New York, NY, USA, 2005, WWW '05, pp. 1072–1073, ACM.

[18] Hyun-Chul Kim, Shaoning Pang, Hong-Mo Je, Daijin Kim, and Sung Yang Bang, "Pattern classification using support vector machine ensemble," *Pattern Recognition, International Conference on*, vol. 2, pp. 20160, 2002.

[19] Liyun Li, Umut Topkara, Baris Coskun, and Nasir Memon, "Cocost: A computational cost sensitive classifier," in *ICDM'09: IEEE International Conference on Data Mining, Miami FL*, 2009.

[20] Mehdi Kharrazi, Husrev T. Sencar, and Nasir Memon, “Improving steganalysis by fusion techniques: A case study with image steganography.,” *LNCS Transactions on Data Hiding and Multimedia Security I*, vol. 4300, pp. 123–137, 2006.

[21] Mehdi Kharrazi, Husrev T. Sencar, and Nasir Memon, “Performance study of common image steganography and steganalysis techniques,” *Journal of Electronic Imaging*, vol. 15, no. 4, 2006.

[22] Tomás Pevný and Jessica J. Fridrich, “Multiclass detector of current steganographic methods for jpeg format,” *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 4, pp. 635–650, 2008.

[23] T. Pevný and J. Fridrich, “Multiclass blind steganalyasis for JPEG images,” in *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VIII, San Jose, CA, January 16–19*, E. Delp III and P. W. Wong, Eds., January 2006, vol. 6072, pp. 1–13.

[24] J. R. Quinlan, “Bagging, boosting, and c4.5,” in *In Proceedings of the Thirteenth National Conference on Artificial Intelligence*. 1996, pp. 725–730, AAAI Press.

[25] Leo Breiman, Jerome Friedman, Charles J. Stone, and R. A. Olshen, *Classification and Regression Trees*, Chapman & Hall/CRC, 1 edition, Jan. 1984.

[26] Alex M. Andrew, “An introduction to support vector machines and other kernel-based learning methods,” *Robotica*, vol. 18, pp. 687–689, November 2000.

[27] Thomas G. Dietterich, “An experimental comparison of three methods for constructing ensembles of decision trees,” in *Bagging, boosting, and randomization. Machine Learning*, 2000, pp. 139–157.

[28] E.Schapire Leo Breiman, “Random forest,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, October 2001.

[29] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore, “Reinforcement learning: A survey,” *JOURNAL OF ARTIFICIAL INTELLIGENCE RESEARCH*, vol. 4, pp. 237–285, 1996.

[30] Jeffrey C. Schlimmer Ming Tan, “Two case studies in cost-sensitive concept acquisition,” in *In Proceedings of the Eighth National Conference on Artificial Intelligence*, 1990.

[31] Marlon Nunez, “The use of background knowledge in decision tree induction,” *Machine Learning*, vol. 6, no. 3, pp. 231–250, May 1991.

[32] Antonio Torralba, Kevin P. Murphy, and William T. Freeman, “Sharing features: efficient boosting procedures for multiclass object detection,” in *In CVPR*, 2004, pp. 762–769.

[33] T. Pevný and J. Fridrich, “Benchmarking for steganography,” in *Information Hiding, 10th International Workshop*, K. Solanki, Ed., Santa Barbara, CA, May 19–21, 2008, Lecture Notes in Computer Science, Springer-Verlag, New York.

[34] T. Pevny and J. Fridrich, “Merging Markov and DCT features for multi-class JPEG steganalysis,” *Security, Steganography, and Watermarking of Multimedia Contents IX*, pp. 1–13, 2007.

[35] Q. Liu, A.H. Sung, and M. Qiao, “Improved detection and evaluation for JPEG steganalysis,” in *Proceedings of the seventeen ACM international conference on Multimedia*. ACM, 2009, pp. 873–876.

[36] N. Provos, “Defending against statistical steganalysis,” in *10th USENIX Security Symposium*. Citeseer, 2001, vol. 10, pp. 323–336.

[37] A. Westfeld, “F5-a steganographic algorithm: High capacity despite better steganalysis,” in *Information Hiding: 4th International Workshop, IH 2001, Pittsburgh, PA, USA*, April 25-27 2001.

[38] P. Sallee, “Model-based steganography,” *Lecture Notes in Computer Science*, pp. 154–167, 2003.

[39] Josef Kittler, Mohamad Hatef, Robert P.W. Duin, and Jiri Matas, “On combining classifiers,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 226–239, 1998.

[40] Devi Parikh and Robi Polikar, “An ensemble-based incremental learning approach to data fusion,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 37, no. 2, pp. 437–450, 2007.

[41] Thomas Hancock, Tao Jiang, Ming Li, and John Tromp, “Lower bounds on learning decision lists and trees,” *Information and Computation*, vol. 126, no. 2, pp. 114 – 122, 1996.

[42] Ludmila I. Kuncheva and Christopher J. Whitaker, “Using diversity with three variants of boosting: Aggressive, conservative, and inverse,” *Lecture Notes in Computer Science*, pp. 81–90, 2002.

[43] Y. Freund and R.E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” *Journal of Computer and System Sciences*, vol. 55, pp. 119–139, 1997.

[44] T. Pevny and J. Fridrich, “Merging Markov and DCT features for multi-class JPEG steganalysis,” *Security, Steganography, and Watermarking of Multimedia Contents IX*, pp. 1–13, 2007.

[45] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM: a library for support vector machines*, 2001, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[46] Gokhan Gul, Ahmet Emir Dirik, and Ismail Avcibas, “Steganalytic features for jpeg compression-based perturbed quantization,” *IEEE Signal Processing Letters*, 2000.